

МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
ДЕПАРТАМЕНТ НАУЧНО-ТЕХНОЛОГИЧЕСКОЙ ПОЛИТИКИ И
ОБРАЗОВАНИЯ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«КРАСНОЯРСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

**Разработка бизнес-приложений
на платформе
«1С: Предприятие 8»**

Красноярск
2018

Разработка бизнес-приложений на платформе «1С: Предприятие 8» [Текст]: методические указания к выполнению лабораторных работ / сост.: Миндалёв И.В. -- Красноярск, Краснояр. гос. аграр. ун-т., 2018, 181 с.

Методические указания представляют собой руководство по разработке бизнес-приложений на платформе «1С: Предприятие 8».

Предназначено для студентов, изучающих дисциплины «Электронный бизнес», «Информационные системы управления производственной компанией», «Практика по получению профессиональных умений и опыта профессиональной деятельности», «Преддипломная практика» по направлению 38.03.05 (5.38.03.05) «Бизнес-информатика», «Автоматизированные системы бухгалтерского учета в агропромышленном комплексе», «Теория экономических информационных систем», «Предметно-ориентированные экономические информационные системы», «Практика по получению профессиональных умений и опыта профессиональной деятельности», «Преддипломная практика» по направлению 09.03.03 (2.09.03.03) «Прикладная информатика», «Автоматизированные системы бухгалтерского учёта», «Разработка информационных систем на платформе 1С», «Принципы построения и функционирования экономических информационных систем на платформе 1С» по направлению 01.03.02 (1.01.03.02) Прикладная математика и информатика.

© Красноярский государственный аграрный университет, 2018

© Миндалёв И.В., 2018

1-й день. Начало

Психология рук, вооруженных инструментами, должна входить в структуру личности. У всякого инструмента есть коэффициент доблести и коэффициент сообразительности. Для доблестного рабочего инструмент обладает смыслом.

Г. Башляр. Режущая воля и твердые материалы

1.1. Установка платформы

1. Запустите установку программ: в окне **Проводника** откройте каталог **1с82_prog**, выберите **autorun.exe**, нажмите **Enter**.

2. Запустите установку платформы 1С:Предприятие 8.0: выберите **Выборочная установка, 1С: Предприятие 8.2 (учебная версия)**.

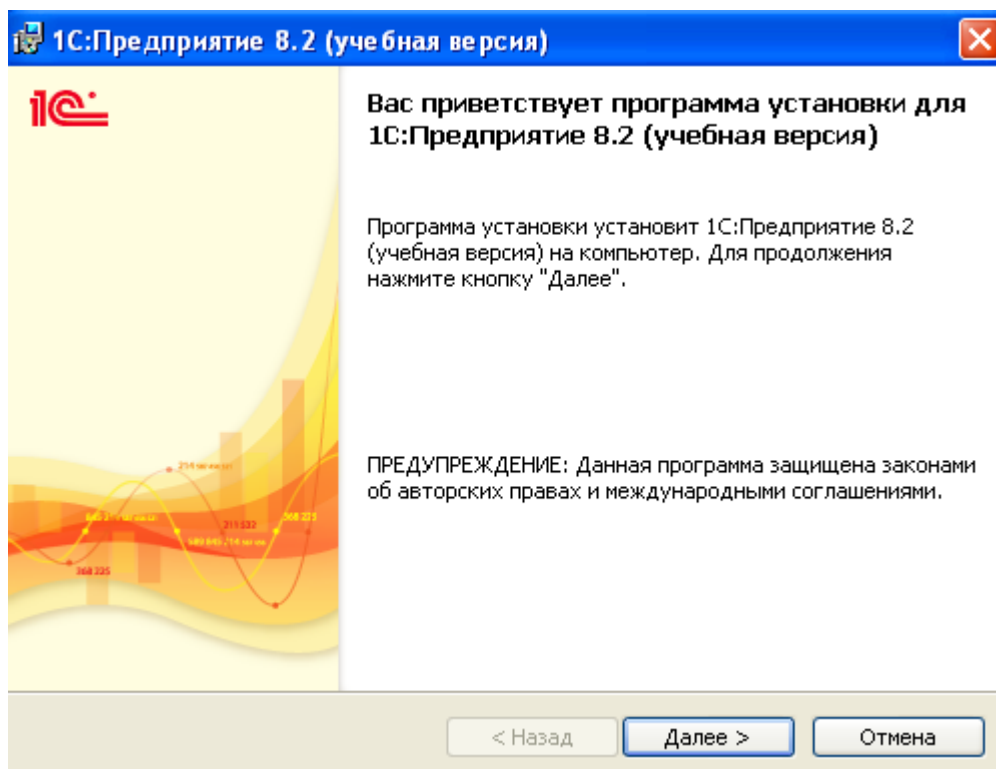


Рис. 1:

2. В окне установки выберите **Далее**.

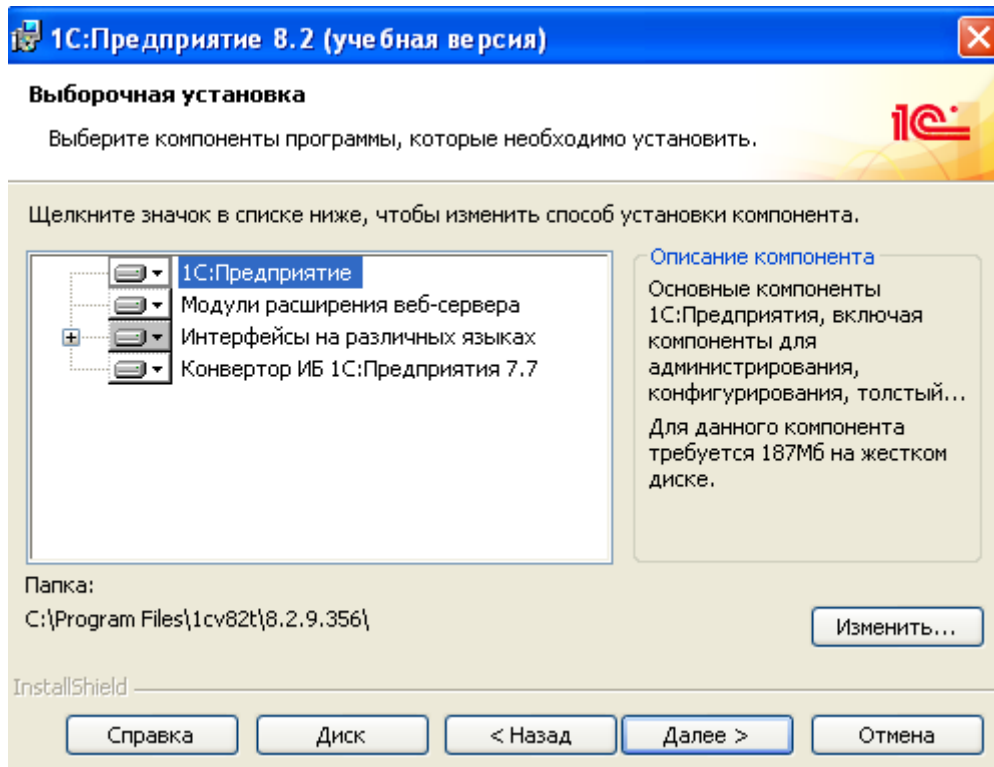


Рис. 2:

3. В окне Выборочная установка выберите **Далее**.

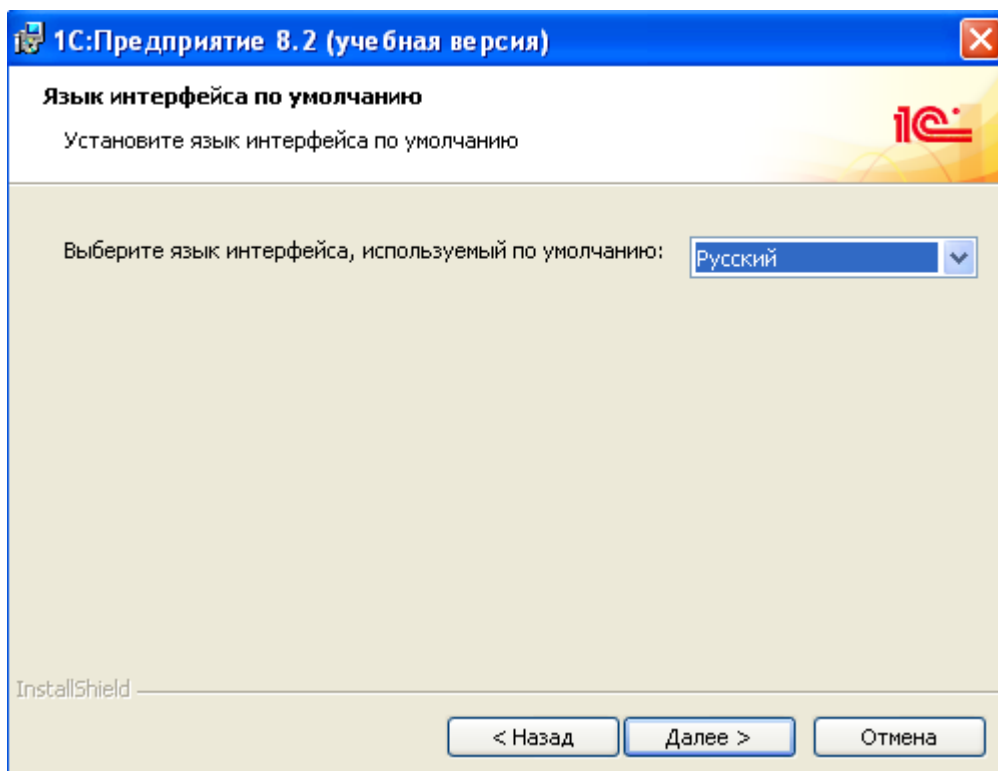


Рис. 3:

4. В окне Язык интерфейса по умолчанию выберите **Русский**, нажмите **Далее**.

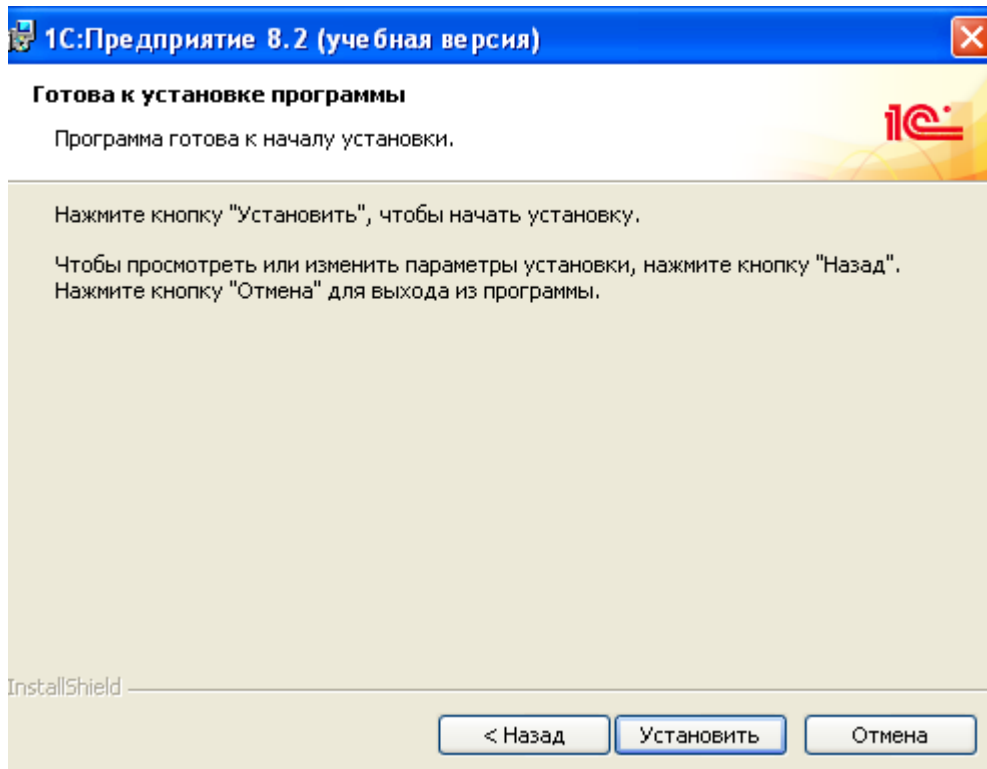


Рис. 4:

5. В окне установки нажмите **Установить**.

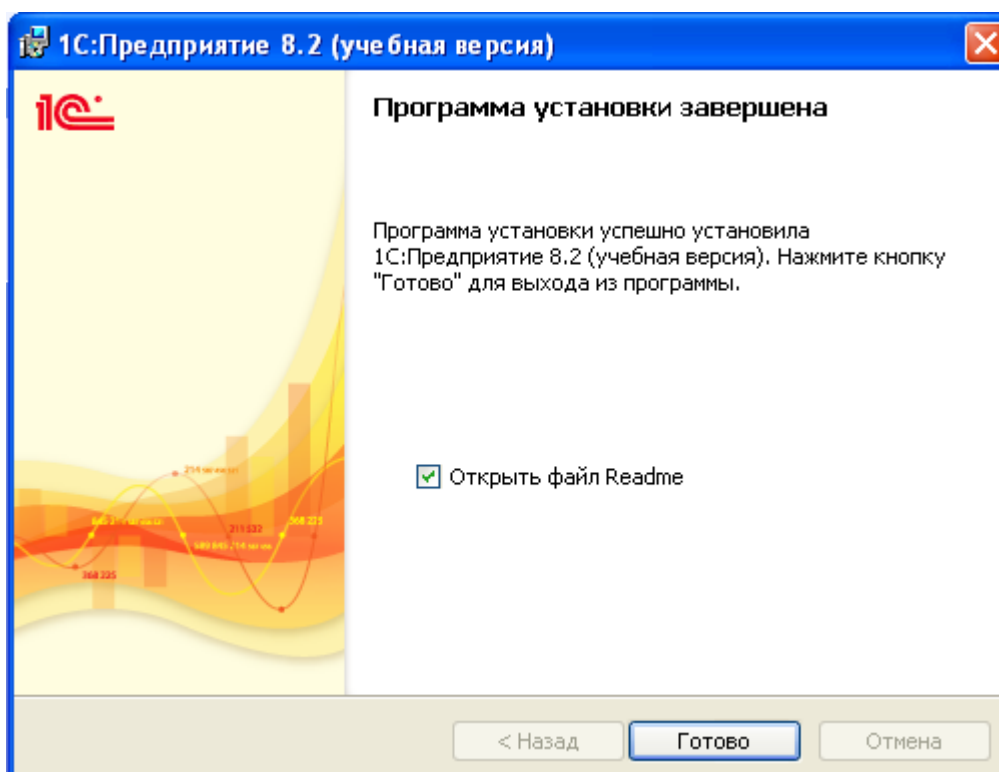


Рис. 5:

6. В окне установки нажмите **Готово**.

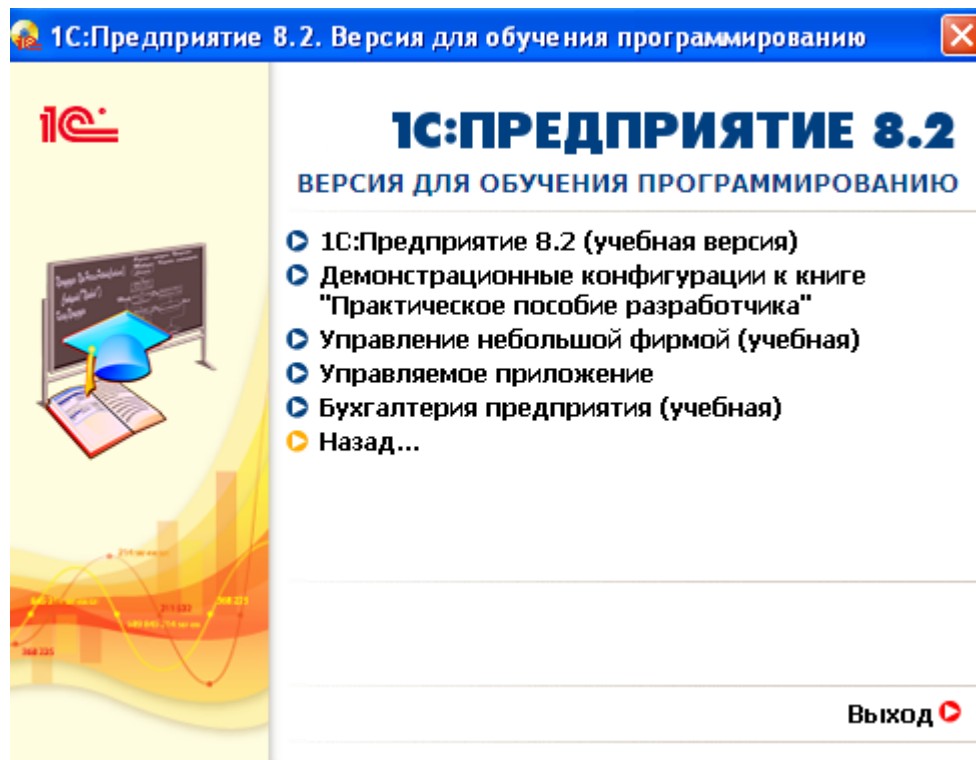
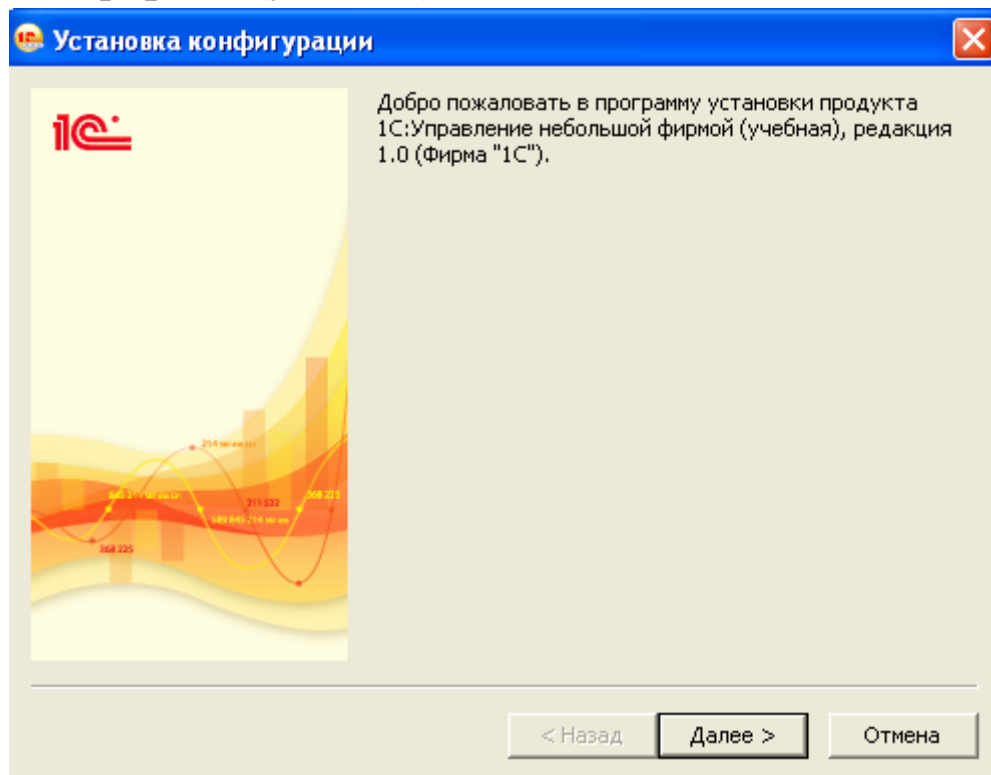


Рис. 6:

7. Запустите установку конфигурации: выбери **Управление небольшой фирмой (учебная)**.



8. В окне установки выберите **Далее**.

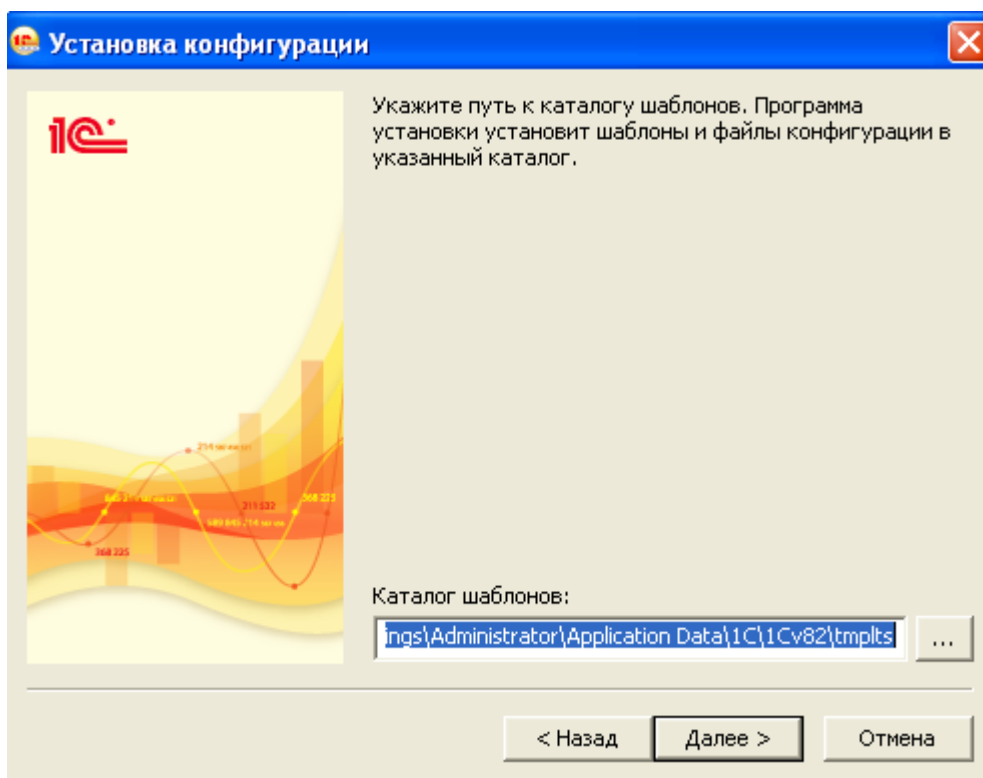


Рис. 7:

Программа установки типовой конфигурации фактически устанавливает шаблоны информационных баз. Каталог установки шаблонов можно изменить, но это должен быть подкаталог `\tplts` каталога самой программы (например, `C:\Program Files\1cv8\tmplts`), в противном случае для возможности создания информационных баз по шаблону в дальнейшем необходимо будет указать нужный каталог в диалоге «Запуск 1С:Предприятия» с помощью кнопки «Настройка...».

9. Выберите путь к каталогу шаблонов предлагаемый по умолчанию: нажмите **Далее**.

10. Установка закончена: нажмите **Готово**.

11. Запустите установку конфигурации: выбери **Бухгалтерия предприятия (учебная)**.

1.2. Программирование или разработка

Что будем делать через 5 минут? Писать программу на 1С. Но есть другие вопросы. На чем вы работаете? На 1С. На чем это написано? На 1С. Требуется бухгалтер со знанием 1С, требуется программист 1С на неполный рабочий день. Слишком много 1С. Поэтому на вопрос, что будем делать через 5 минут? Отвечаем: будем изучать мето-

ды разработки прикладных решений на основе платформы 1С:Предприятия 8.2

1.3. Общие сведения о системе 1С:Предприятие

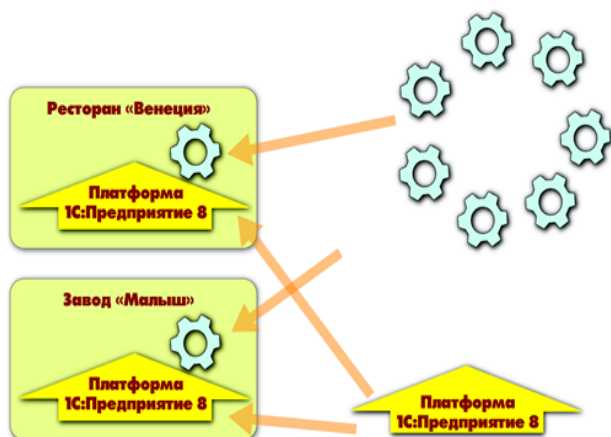


Рис. 1.1. Конфигураций много, а платформа – одна

Система 1С:Предприятие является универсальной системой автоматизации экономической и организационной деятельности предприятия. Поскольку такая деятельность может быть довольно разнообразной, система 1С:Предприятие имеет возможность «приспосабливаться» к особенностям конкретной области деятельности, в которой она используется. Для обозначения такой способности используется термин **конфигурируемость**, то есть возможность настройки системы на особенности конкретного предприятия и класса решаемых задач.

Это достигается тем, что 1С:Предприятие - это не просто программа, существующая в виде набора неизменяемых файлов, а совокупность различных программных инструментов, с которыми работают разработчики и пользователи. Логически всю систему можно разделить на две большие части, которые тесно взаимодействуют друг с другом: **конфигурацию** и **платформу**, которая управляет работой конфигурации.

Для того чтобы легче понять взаимодействие этих частей системы, сравним ее с проигрывателем компакт-дисков. Как вы хорошо знаете, проигрыватель служит для того чтобы слушать музыку. «На вкус и цвет товарищей нет», поэтому существует множество разнообразных компакт-дисков, на которых записаны музыкальные произведения на любой вкус. И для того, чтобы прослушать какую либо композицию, нужно вставить компакт-диск в проигрыватель, и

проигрыватель воспроизведет записанное на нем музыкальное произведение. Более того, современный проигрыватель компакт-дисков даже позволит вам записать собственную подборку музыкальных произведений, т.е. создать новый компакт-диск.

Сам по себе проигрыватель совершенно бесполезен без компакт-диска, точно так же, как компакт-диск не может сам по себе принести нам никакой пользы (кроме как стать подставкой под чашку кофе ©), если у нас нет проигрывателя.

Возвращаясь к системе 1С:Предприятие, можно сказать, что платформа является своеобразным «проигрывателем», а конфигурация - «компакт-диском». Платформа обеспечивает работу конфигурации и позволяет вносить в нее изменения или создавать собственную конфигурацию.

Существует одна платформа (1С:Предприятие 8.0) и множество конфигураций. Для функционирования какого-либо прикладного решения всегда необходима платформа и какая-либо (одна) конфигурация.

Сама по себе платформа не может выполнить никаких задач автоматизации, так как она создана для обеспечения работы какой-либо конфигурации. То же самое с конфигурацией: чтобы выполнить те задачи, для которых она создана, необходимо наличие платформы, которая и управляет ее работой.

1.4. Конфигурация и прикладное решение

Здесь следует сказать о небольшой двойственности терминологии, которая будет использоваться в дальнейшем. Двойственность заключается в употреблении разных терминов для обозначения одного и того же предмета: конфигурация и прикладное решение. Оба эти термина обозначают ту часть системы 1С:Предприятие, которая работает под управлением платформы и которую «видят» все пользователи (бывает, конечно, что пользователи работают и с инструментальными средствами платформы, но это «продвинутые» пользователи). Употребление одного или другого термина зависит от контекста, в котором ведется изложение.

Если речь идет о действиях разработчика, то употребляется термин конфигурация, поскольку это точный термин 1С:Предприятия.

Термин прикладное решение является более общепринятым и понятным для пользователя системы 1С:Предприятие.

Итак, поскольку задачи автоматизации, как было упомянуто выше, могут быть самыми разными, фирма 1С и ее партнеры выпускают прикладные решения, каждое из которых предназначено для автоматизации одной определенной области человеческой деятельности.

В качестве примера существующих прикладных решений можно перечислить следующие типовые решения:

1С:Бухгалтерия 8 — универсальная программа массового назначения для автоматизации бухгалтерского и налогового учета, включая подготовку обязательной (регламентированной) отчетности. Это готовое решение для ведения учета в организациях, осуществляющих любые виды коммерческой деятельности: оптовую и розничную торговлю, комиссионную торговлю (в том числе субкомиссию), оказание услуг, производство и т.д.

1С:Предприятие 8. Управление торговлей — в комплексе решает задачи управленческого и оперативного учета, анализа и планирования; автоматизирует торговые, финансовые и складские операции; обеспечивает современный уровень управления предприятием.

1С:Зарплата и Управление Персоналом 8 — предназначена для комплексной автоматизации расчета заработной платы и реализации кадровой политики предприятий. Это прикладное решение нового поколения, в котором учтены как требования законодательства, так и реальная практика работы предприятий, а также перспективные мировые тенденции развития подходов к мотивации и управлению персоналом.

1С:Предприятие 8. Управление производственным предприятием — является комплексным прикладным решением, охватывающим основные контуры управления и учета на производственном предприятии. Решение позволяет организовать комплексную информационную систему, соответствующую корпоративным, российским и международным стандартам и обеспечивающую финансово-хозяйственную деятельность предприятия.

1С:Консолидация 8.0 — программный продукт, предназначенный для решения широкого спектра задач по подготовке и анализу корпоративной отчетности групп компаний и филиальных структур в интересах внутренних и внешних потребителей.

Прикладное решение является, по сути, универсальными способно удовлетворить потребности самых разных предприятий, работающих

в одной области деятельности. И это хорошо. С другой стороны, такая универсальность неизбежно приведет к тому, что на конкретном предприятии будут использоваться далеко не все возможности прикладного решения, а каких-то возможностей в нем будет не доставать (нельзя угодить всем).

Вот тут и выходит на передний план конфигурируемость системы, поскольку платформа, помимо управления работы конфигурацией, содержит средства, позволяющие вносить изменения в используемую конфигурацию. Более того, платформа позволяет создать свою собственную конфигурацию «с нуля», если по каким-либо причинам использование типовой конфигурации представляется нецелесообразным.

Обратите внимание, как мы в одном абзаце перешли от прикладного решения к конфигурации. Ничего не поделаешь: для пользователя понятнее так, а для разработчика – по-другому.

Таким образом, если вернуться к сравнению с проигрывателем компакт-дисков, мы можем изменять по своему вкусу мелодии, которые были ранее записаны на компакт-диске, и даже создавать диски со своими собственными музыкальными произведениями. При этом нам не потребуются какие-либо музыкальные инструменты – все необходимое для создания мелодий есть в нашем проигрывателе компакт-дисков.

1.5. Режимы работы системы

Для того чтобы обеспечить такие возможности, система 1С:Предприятие имеет различные режимы работы: 1С:Предприятие и Конфигуратор.

Режим 1С:Предприятие является основным и служит для работы пользователей системы. В этом режиме пользователи вносят данные, обрабатывают их и получают выходные результаты.

Режим конфигуратора используется разработчиками и администраторами информационных баз. Именно этот режим и предоставляет инструменты, необходимые для модификации существующей или создания новой конфигурации.

Поскольку задача нашей книги состоит в том, чтобы научить вас создавать собственные конфигурации и изменять существующие, дальнейшее повествование будет в основном посвящено работе с системой в режиме конфигуратора. И лишь иногда, чтобы проверить ре-

зультаты нашей работы, мы будем запускать систему в режиме 1С:Предприятие.

1.6. Создание новой ИБ

1. Запустите конфигуратор: выберите **Пуск, Программы, 1С:Предприятие 8.2** (учебная версия), **1С:Предприятие**

2. Создайте пустую ИБ: выберите **Добавить**, появится окно, в котором предлагается либо создать новую ИБ, либо подключить уже имеющуюся, выберите **Создание новой информационной базы**, нажмите **Далее**, выберите **Создать информационной базы без конфигурации для разработки или для загрузки выгруженной ранее информационной базы**, **Далее**.

3. В следующем окне предлагается указать название ИБ и выбрать место ее расположения: на данном локальном компьютере, другом компьютере сети или на сервере. В нашем случае мы будем работать с ИБ, размещенной на локальном компьютере. Выберите **на данном компьютере или на компьютере в локальной сети**, введите имя ИБ **Посад_Зайцева** (где вместо Зайцевой введите свою фамилию), нажмите **Далее**.

4. Программа предлагает указать место расположения ИБ на компьютере: выберите **многоточие**, откройте каталог **info_base**, создайте каталог **posad_zauseva**, нажмите **Открыть**, нажмите **Далее**.

5. Укажите параметры запуска: **вариант аутентификации** выберите **Выбирать автоматически**, **основной режим запуска** выберите **Выбирать автоматически**, нажмите **Готово**.

6. Запустите конфигуратор : выберите **Посад_Зайцевой**, выберите **Конфигуратор**.

7. Откройте окно: выберите **Конфигурация, Открыть конфигурацию**.

1.7. Дерево объектов конфигурации

Можно сказать, что дерево конфигурации – основной инструмент, с которым работает разработчик. Дерево конфигурации содержит в себе практически всю информацию о том, из чего состоит конфигурация.

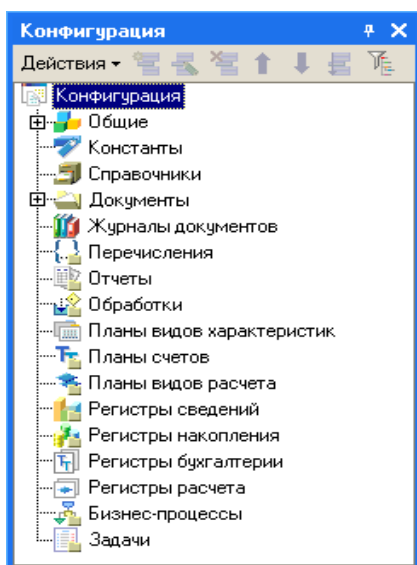


Рис. 8: Дерево конфигурации

Рис. 1.3.

Наверняка у вас уже возник вопрос: почему в дереве что-то есть, если мы пока еще ничего не создавали? Дело в том, что для облегчения работы разработчика «все, из чего состоит конфигурация» сгруппировано, и сейчас дерево и показывает вам эти группы.

Если вы походите по дереву и понажимаете на плюсики, то увидите, что ни в одной группе ничего нет. Исключение составит лишь группа Общие, Языки, в которой вы обнаружите «нечто» под названием «Русский». Этот «Русский» платформа создала для вас сама, поскольку в данном случае конфигуратор использует русскоязычный интерфейс.

Хотелось бы уже начать что-нибудь делать, но прежде следует определиться с терминами. Вы наверняка уже заметили, что, говоря о содержимом конфигурации, мы сознательно избегали использования каких-либо терминов. Но теперь настало время, когда можно определиться с терминологией и рассказать про...

1.8. Объекты конфигурации

Конфигурация представляет собой описание. Она описывает структуру данных, которые пользователь будет использовать в режиме работы 1С:Предприятие. Кроме этого конфигурация описывает всевозможные алгоритмы обработки этих данных, содержит информацию о том, как эти данные должны будут выглядеть на экране и на принтере, и т.д.

В дальнейшем платформа 1С:Предприятия на основании этого описания создаст базу данных, которая будет иметь необходимую структуру, и предоставит пользователю возможность работать с этой базой данных.

Для того чтобы систему 1С:Предприятие можно было быстро и легко настраивать на нужные прикладные задачи, все описание, которое содержит конфигурация, состоит из неких логических единиц, называемых объектами конфигурации.

Возможно, вы уже успели заглянуть в книгу 1С:Предприятие 8.2 Конфигурирование и администрирование, в которой дается краткое описание объекта конфигурации.

Мы не будем дублировать это определение в настоящей книге, поскольку наша задача – не изложить концепцию построения системы 1С:Предприятие как структуры метаданных, описанной в терминах классов проблемно-ориентированных бизнес-сущностей, а научить вас методически правильно и грамотно использовать возможности 1С:Предприятия.

Поэтому, что представляют собой объекты конфигурации, мы объясним на «бытовом» уровне. Однако он даст вам возможность правильно понимать назначение объектов применительно к тем задачам, которые мы будем решать.

С одной стороны, объекты конфигурации представляют собой детали конструктора, из которого собирается конфигурация. Обычно в конструкторе существует некоторый набор деталей. Детали могут быть разного вида: длинные, короткие, квадратные, прямоугольные и т.д. Теперь представьте, что деталей каждого вида мы можем создавать столько, сколько нам нужно (скажем, 5 длинных и 3 коротких). Мы можем соединять детали между собой различными способами.

То же и с объектами конфигурации. Мы можем создавать только объекты определенных видов. Но каждого вида объектов мы можем создать столько, сколько нам нужно. Объекты одного вида отличаются от объектов другого вида тем, что имеют разные свойства (точнее говоря, разный набор свойств). Объекты могут взаимодействовать друг с другом, и мы можем описать такое взаимодействие.

В чем еще сходство объектов конфигурации с деталями конструктора? В конструкторе обычно есть блоки, которые можно скрепить между собой, и есть другие детали, например колеса, которые скрепить между собой нельзя, зато их можно соединить с осью, и тогда

колеса будут вращаться. Т.е. разные детали конструктора по-разному ведут себя.

Объекты конфигурации также обладают различным поведением, и оно зависит от вида объекта. Одни объекты могут выполнять какие-то действия, другие этих действий выполнять не могут, зато у них есть свой собственный набор действий.

Следующую особенность объектов конфигурации можно продемонстрировать на примере автомобиля. Автомобиль состоит из большого количества деталей. Одна из деталей автомобиля – это двигатель. Но двигатель, в свою очередь, тоже состоит из набора деталей, причем в разных двигателях могут использоваться одни и те же детали.

Так же «сложные» объекты конфигурации состоят из более «простых», и одни и те же «простые» объекты могут входить в состав сложных объектов. Такая структура позволяет упростить работу с объектами конфигурации, поскольку если мы знаем, как работать с каким-либо «простым» объектом, то в любом «сложном» объекте, в состав которого он входит, мы будем работать с ним все тем же образом.

И, наконец, самое важное качество объектов конфигурации – это их прикладная направленность. Объекты конфигурации не просто некие абстрактные конструкции, при помощи которых разработчик пытается описать поставленную перед ним задачу. Они представляют собой аналоги реальных объектов, которыми оперирует предприятие в ходе своей работы.

Например, на каждом предприятии существуют различные документы, с помощью которых оно фиксирует факты совершения хозяйственных операций. Точно так же в конфигурации существуют объекты вида «Документ».

Кроме этого на каждом предприятии обязательно ведется список сотрудников, справочник номенклатуры или товаров. В конфигурации тоже есть специальные объекты вида «Справочник», которые позволяют разработчику создавать компьютерные аналоги таких списков.

На основе объектов конфигурации платформа создает в базе данных информационные структуры, в которых будут храниться данные. В литературе, как правило, объект конфигурации и соответствующую ему информационную структуру принято называть одинаково.

Например, если в конфигурации существует объект Справочник Сотрудники, то информационную структуру, созданную платформой на основе этого объекта конфигурации, также называют справочником Сотрудники.

Мы отойдем от такого «размытого» стиля изложения и в тех местах, где речь пойдет о конфигурации, будем использовать явное уточнение – **объект конфигурации Справочник Сотрудники**. Там же, где речь пойдет о базе данных, мы будем говорить просто: **справочник Сотрудники**.

8. Зададим имя конфигурации: выделим в дереве объектов конфигурации корневой элемент **Конфигурация, М2**, в палитры свойств в поле **Имя** введите **Посад**.

9. Проверим наши изменения в режиме 1С:Предприятие: выберите **Отладка, Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**.

На экране появится окно 1С:Предприятие. В заголовке окна мы видим название нашей конфигурации: **Посад**. Пустое пространство – это рабочая область приложения, которая пока ничем не заполнена. Мы не создали никаких объектов конфигурации и не создали никаких подсистем, в которых бы эти объекты отображались.

1.9. Что такое подсистема

Платформа 1С:Предприятия 8 позволяет выделить в прикладном решении отдельные части, - подсистемы, - в совокупности представляющие все прикладное решение. Подсистемы могут иметь иерархическую структуру, т.е. одна подсистема может включать в себя несколько других подсистем:

Для каждого объекта конфигурации существует возможность указать его принадлежность к одной или нескольким подсистемам. Таким образом, в терминах подсистем можно описать всю структуру прикладного решения:

В дальнейшем это описание может быть использовано для облегчения труда разработчика. Например, информацию, отображаемую в окне конфигурации можно отбирать по принадлежности к какой-либо подсистеме, и таким образом оперировать не всеми объектами конфигурации, а только теми,

которые имеют отношение к разрабатываемой в данный момент части прикладного решения.

Еще одним применением механизма подсистем является возможность автоматического формирования прав на основе подсистем, и возможность автоматического построения интерфейсов пользователей на их основе.

При описании прав пользователей существует возможность установить или снять права только для тех объектов прикладного решения, которые относятся к указанным подсистемам:

При создании пользовательских интерфейсов существует возможность автоматически создать интерфейс, включающий в себя команды для работы с объектами, относящимися к одной или нескольким выбранным подсистемам:

1.10. Добавление подсистемы

10. Закроем окно 1С:Предприятие.

11. Создадим новую подсистему: в дереве объектов конфигурации раскройте ветвь **Общие**, выделим ветвь **Подсистемы**, **МП**, выберите **Добавить**, в поле **Имя** введите **Бухгалтерия**.

12. Зададим картинку для отображения подсистемы: в поле **Картинка** выберите три точки, выберите вкладку **Из конфигурации**, нажмите **Добавить**, в поле **Имя** введите **Бухгалтерия**, выберите **Выбрать из файла**, выберите файл **Бухгалтерия.png**, нажмите **Открыть**, закройте окно **Общая картинка**, выберите картинку **Бухгалтерия**, нажмите **ОК**, выберите **Заккрыть**.

13. Создайте подсистемы **УчетМатериалов**, **ОказаниеУслуг**, **РасчетЗарплаты**, **Предприятие**.

14. Проверим наши изменения в режиме 1С:Предприятие: выберите **Отладка**, **Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**.

Вид разрабатываемого приложения изменился. Под главным меню теперь располагается панель разделов приложения, где и отражены созданные нами подсистемы. Разделы представлены в форме гиперссылок, нажав на которые пользователь может открыть связанные с ними документы, справочники, отчеты и т. п.

Раздел Рабочий стол формируется платформой по умолчанию. Он предназначен для размещения наиболее часто используемых пользователем документов, отчетов и т.п.

15. Закройте окно 1С:Предприятие.

16. Изменим порядок расположения подсистем: выберите корень дерева объектов конфигурации **Посад, МП**, выберите **Открыть командный интерфейс конфигурации**, с помощью стрелок установите следующий порядок расположения подсистем: **Учет материалов, Оказание услуг, Бухгалтерия, Расчет зарплаты, Предприятие**, нажмите **ОК**.

17. Проверим наши изменения в режиме 1С:Предприятие: выберите **Отладка, Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**.

Порядок расположения подсистем в панели разделов приложения должен измениться.

18. Закройте окно 1С:Предприятие.

19. Создайте копию информационной базы: выберите **Администрирование, Выгрузить информационную базу**, в поле **Имя файла** введите **Посад_1**, нажмите **Сохранить**, после появления сообщения **Выгрузка информационной базы в файл завершена**, нажмите **ОК**.

Контрольные вопросы

- Что такое конфигурируемость системы 1С:Предприятие.
- Из каких основных частей состоит система.
- Что такое платформа и что такое конфигурация.
- Для чего используются разные режимы запуска системы 1С:Предприятие.
- Что такое дерево объектов конфигурации.
- Что такое объекты конфигурации.
- Что создает система на основе объектов конфигурации.
- Какими способами можно добавить новый объект конфигурации.
- Как запустить 1С:Предприятие в режиме отладки.

- Для чего используется объект конфигурации Подсистема.
- Как описать логическую структуру конфигурации при помощи объектов Подсистема.
- Как управлять порядком вывода и отображения подсистем в конфигурации.

2-й день. Справочники

Если камень холоден, то бриллиант горяч.

Г. Башляр. Грезы о кристаллах

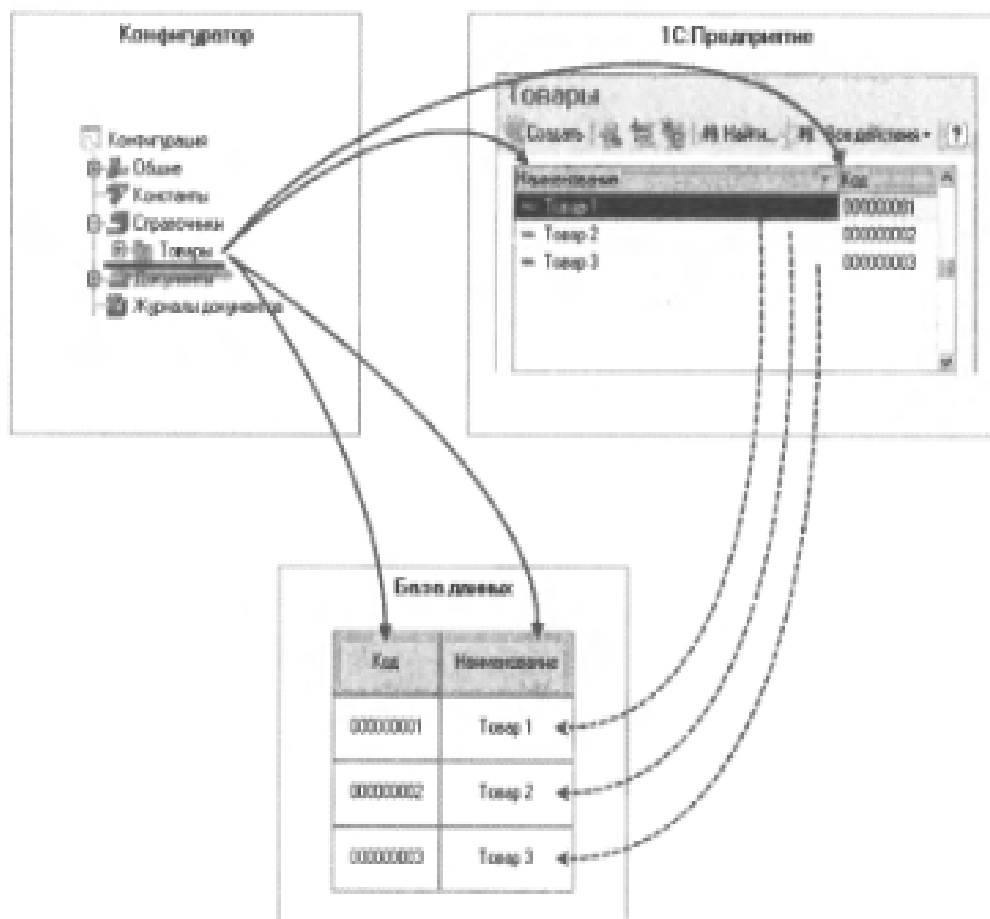
2.1. Что такое справочник

Справочники - это прикладные объекты конфигурации. Они позволяют хранить в информационной базе данные, имеющие одинаковую структуру и списочный характер. Это может быть, например, список сотрудников, перечень товаров, список поставщиков или покупателей.

Справочник состоит из элементов. Например, для справочника сотрудники элементом является сотрудник.

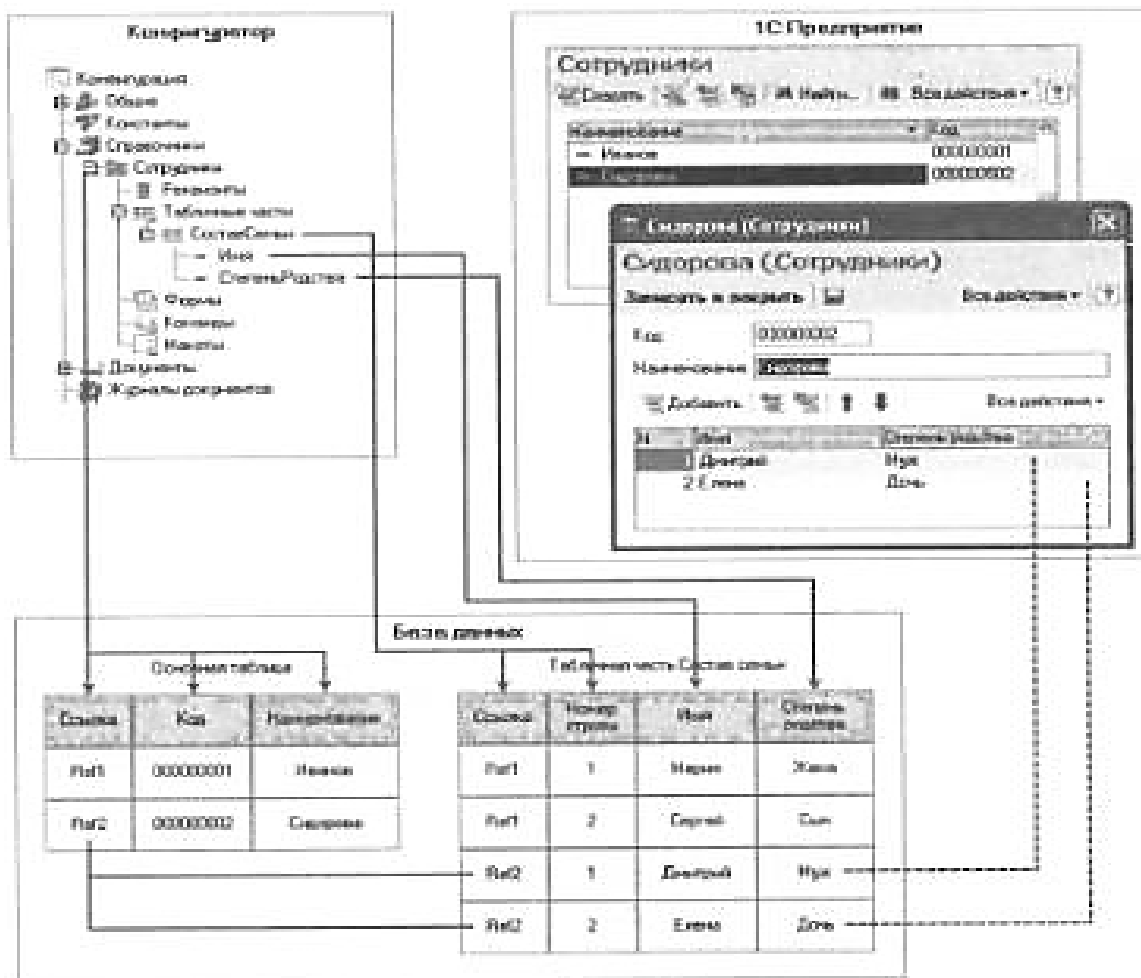
Каждый элемент справочника характеризуется кодом и наименованием. Система поддерживает режим автоматической нумерации элементов, при котором она самостоятельно может генерировать код для нового элемента справочника. Кроме этого система позволяет осуществлять контроль уникальности кодов справочника, не разрешая создавать элементы с одинаковыми кодами.

Помимо кода и наименования, каждый элемент справочника, как правило, содержит некоторую дополнительную информацию, которая подробно описывает этот элемент. Например, для товара это может быть информация об артикуле, номере государственной таможенной декларации, стране происхождения и т.п. Набор такой информации является одинаковым для всех элементов конкретного справочника, и для ее хранения служат реквизиты справочника:



В базе данных справочник хранится в виде таблицы, в строках которой расположены элементы списка, а каждому реквизиту в этой таблице соответствует отдельный столбец.

Кроме этого, каждый элемент справочника может содержать некоторый набор информации, которая одинакова по своей структуре, но различна по количеству, для разных элементов справочника. Например, для каждого сотрудника в справочнике **Физические лица** это может быть контактная информация или информация о составе семьи, образовании. Для хранения подобных данных служат табличные части справочника.



2.2. Простой справочник

Итак, вы запутались и хотите вернуться к варианту конфигурации пункт 19.

20. Загрузите информационную базу: выберите **Администрирование**, **Загрузить информационную базу**, выберите **Посад_1**, нажмите **Открыть**, после появления сообщения **После загрузки информационной базы работа Конфигуратора будет завершена. Несохранившиеся данные в открытых окнах могут быть потеряны! Продолжить?** Нажмите **Да**. После сообщения **Информационная база успешно загружена. Работа Конфигуратора будет завершена. Перезапустить Конфигуратор?** Нажмите **Да**.

21. Создайте справочник: выберите на дереве объектов конфигурации ветвь **Справочники**, нажмите **МП**, выберите **Добавить**, на вкладке **Основные** в поле **Имя** введите **Клиенты**, нажмите **tab** и в поле **Синоним** должно появиться **Клиенты** (свойство

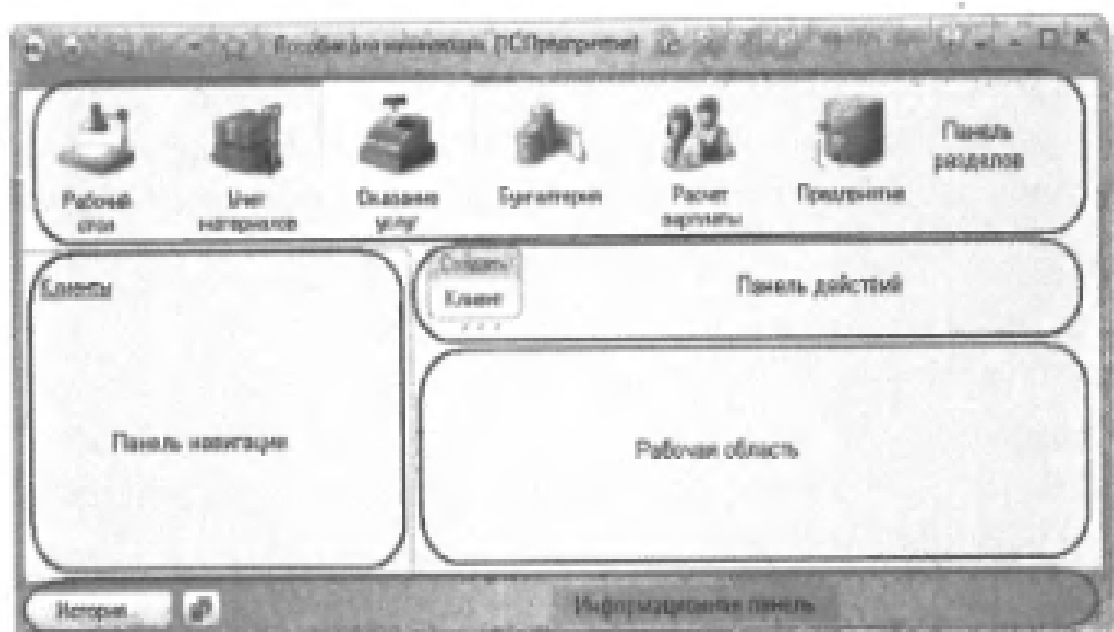
Синоним служит для представления объекта в интерфейсе программы), в поле **Представление объекта** введите **Клиент** (это представление используется для того, чтобы описать, как будет выглядеть в интерфейсе команда добавления нового клиента), в поле **Представление списка** введите **Клиенты**, выберите **Далее**.

22. На вкладке **Подсистемы** выберите **Бухгалтерия** и **ОказанияУслуг**, так оказываемые услуги относятся к определенному клиенту и бухгалтерская отчетность также может быть представлена в разрезе клиентов.

23. Выберите вкладку **Данные** выберите **Длина кода** — **9**, **Длина наименования** — **50**, все остальные свойства оставьте такими, как их предлагает система по умолчанию, нажмите **Заккрыть**.

24. Сделайте доступной в панели действий раздела **ОказаниеУслуг** стандартную команду для создания новых клиентов: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы**, **МП**, **Все подсистемы**, в окне **Все подсистемы** слева в списке **Подсистемы** выберите подсистему **ОказаниеУслуг**, в группе **Панель действий.Создать** для команды **Клиент.Создать** включите галочкой видимость.

25. Проверим наши изменения в режиме **1С:Предприятие**: выберите **Отладка**, **Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне **Реорганизация информации** выберите **Принять**.



Перед нами окно системы в режиме 1С:Предприятие. Если перейти в раздел Оказание услуг или бухгалтерия, то слева в вертикальной области окна появится **панель навигации**.

Сейчас она содержит команду для открытия списка – Клиенты.

Панель навигации отображает структуру выбранного раздела. И предназначена для быстрого перехода к различным спискам в пределах выбранного раздела.

Также появилась **панель действий**. Она содержит команды, которые соответствуют текущему разделу, выбранному в панели разделов.

Сейчас в панели действий раздела Оказание услуг в группе Создать доступна команда для создания элементов нашего справочника клиенты.

26. Создайте элемент справочника: выберите раздел **Оказание услуг**, в панели действий выберите **Клиент**, в поле **Код** данные вносим не будем так он генерируется автоматически, в поле **Наименование** введите **Зырянов Алексей Владимирович**, нажмите **Записать и закрыть**.

27. Введите еще двух клиентов **Василихина Ольга Георгиевна**, **Арляпов Антон Андреевич**.

Обратите внимание, что поле Наименование при вводе нового клиента подсвечено красным пунктиром. Это значит, что для этого поля по умолчанию выполняется проверка заполнения. Если это поле оставить пустым и попытаться записать клиента, то будет выдано сообщение об ошибке.

28. Закройте окно 1С:Предприятие.

29. Проверьте заполнение стандартных реквизитов: в режиме конфигуратора откройте справочник **Клиенты**, выберите вкладку **Данные**, **Стандартные реквизиты**, выберите реквизит **Наименование**, **МП**, **Свойства**, в палитре свойств реквизита Наименование видим, что свойство **Проверка заполнения** по умолчанию установлено в значении **Выдавать ошибку**.

2.3. Справочник с табличной частью

Справочник Сотрудники будет устроен несколько сложнее, чем справочник Клиенты. Дело в том, что в нем мы будем хранить не только ФИО сотрудника, но и информацию о его прошлой

трудовой деятельности. Эта информация однородна по своей структуре (организация, начало, окончание работы, занимаемая должность), но количество предыдущих мест работы у разных сотрудников может быть различным. Поэтому для хранения такой информации мы будем использовать табличную часть справочника.

30. Создайте справочник: выберите на дереве объектов конфигурации ветвь **Справочники**, нажмите **МП**, выберите **Добавить**, на вкладке **Основные** в поле **Имя** введите **Сотрудники**, нажмите **tab** и в поле **Синоним** должно появиться **Сотрудники**, в поле **Представление объекта** введите **Сотрудник**, в поле **Расширенное представление списка** введите **Список сотрудников**, выберите **Далее**.

31. На вкладке подсистемы выберите **Расчет зарплаты и Оказания услуг**, так при оказании услуг должен быть указан сотрудник, оказавший эти услуги, и по результатам этой работы мы будем начислять зарплату каждому сотруднику.

32. Выберите вкладку **Данные** выберите **Длина кода — 9**, **Длина наименования — 50**,

33. Создайте новую табличную часть: нажмите **Добавить табличную часть**, в поле **Имя** введите **ТрудоваяДеятельность**, нажмите галочку **Сохранить**.

34. Создайте реквизиты табличной части: выберите табличную часть **ТрудоваяДеятельность**, нажмите **Добавить реквизит** над списком табличных частей справочника, в поле **Имя** введите **Организация**, в поле **Тип данных — Строка**, **Длина — 100**, нажмите галочку **Сохранить**.

35. Создайте реквизиты табличной части **ТрудоваяДеятельность**: **НачалоРаботы — тип Дата**, состав даты — **Дата**, **ОкончаниеРаботы — тип Дата**, состав даты — **Дата**, **Должность — тип Строка**, длина **100**.

36. Сделайте доступной в панели действий раздела **Расчет зарплаты** стандартную команду для создания новых сотрудников: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы**, **МП**, **Все подсистемы**, в окне **Все подсистемы** слева в списке **Подсистемы** выберите подсистему **РасчетЗарплаты**, в группе **Панель действий**.Создать для команды **Сотрудник.создать** включите галочкой видимость.

37. Проверим наши изменения в режиме 1С:Предприятие: выберите **Отладка**, **Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

38. Создайте элементы справочника: выберите раздел **Расчет зарплаты**, в панели действий выберите **Сотрудник**, введите следующие данные:

Голендова Марина Евгеньевна

Трудовая деятельность

Организация ЗАО НТЦ

Начало работы 01.02.2000

Окончание работы 16.04.2003

Должность Ведущий специалист

Пинчук Денис Александрович

Трудовая деятельность

1

Организация ООО Автоматизация

Начало работы 22.01.1996

Окончание работы 31.12.2002

Должность Инженер

2

Организация ЗАО НПО СпецСвязь

Начало работы 20.06.1986

Окончание работы 21.01.1995

Должность Начальник производства

Рогов Денис Владимирович

Трудовая деятельность

Организация ООО СтройМастер

Начало работы 06.02.2001

Окончание работы 03.04.2004

Должность Прораб

2.4. Иерархический справочник

Справочник **Номенклатура** будет содержать информацию об услугах, которые оказывает ООО «На все руки мастер», и о тех материалах, которые при этом могут быть использованы.

Этот справочник не будет сложным Единственная особенность, которой он будет обладать, – это наличие иерархической структуры. Для того чтобы справочником было удобно пользоваться, мы сгруппируем услуги в одну группу, а материалы – в другую. Кроме этого, поскольку ООО «На все руки мастер» оказывает самые разные услуги, они также будут логически собраны в несколько групп. То же самое можно сказать и про материалы.

39. Создайте справочник: выберите на дереве объектов конфигурации ветвь **Справочники**, **МП**, выберите **Добавить**, на вкладке **Основные** в поле **Имя** введите **Номенклатура**, нажмите **tab** и в поле **Синоним** должно появиться **Номенклатура**, так как понятие номенклатура не имеет единственного числа, больше никаких свойств, определяющих представление объекта в интерфейсе приложения, задавать не будем. выберите **Далее**.

40. На вкладке подсистемы выберите **Учет материалов**, **Оказание услуг** и **Бухгалтерия**. К первым двум разделам справочник имеет прямое отношение, а для бухгалтерского анализа всегда может понадобиться список материалов и услуг.

41. Выберите вкладку **Иерархия**, установите флажок **Иерархический справочник**.

42. Выберите вкладку **Данные** выберите **Длина кода — 9**, **Длина наименования — 100**,

43. Сделайте доступной в панели действий раздела **Учет материалов** и **Оказание услуг** стандартную команду для создания новых элементов списка номенклатура: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы**, **МП**, **Все подсистемы**, в окне **Все подсистемы** слева в списке **Подсистемы** выберите подсистему **УчетМатериалов**, в группе **Панель действий**.Создать для команды **Номенклатура.создать** включите галочкой видимость.

44. То же и для раздела **Оказание услуг**: см. 43.

45. Проверим наши изменения в режиме **1С:Предприятие**: выберите **Отладка**, **Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне **Реорганизация информации** выберите **Принять**.

46. Создайте элементы справочника: выберите раздел **Учет материалов**, в панели навигации выберите **Номенклатура**.

47. Создайте две группы **Материалы** и **Услуги** в корне справочника с помощью кнопки **Создать новую группу**.

48. В группе **Материалы** создайте пять элементов с помощью кнопки **Создать**: **Строчный трансформатор Samsung**, **Строчный трансформатор GoldStar**, **Транзистор Philips 2N2369**, **Шланг резиновый**, **Кабель электрический**.

49. В группе **Услуги** создайте элементы: **Диагностика**, **Ремонт отечественного телевизора**, **Ремонт импортного телевизора**, **Подключение воды**, **Подключение электричества**.

50. В группе **Услуги** создайте еще две группы: **Телевизоры** и **Стиральные машины**.

51. Переместить услуги в соответствующие группы.

52. В группе **Материалы** создайте две группы: **Радиодетали** и **Прочее**. В группу **Прочее** поместите **Кабель электрический** и **Шланг резиновый**. Остальные материалы переместите в группу **Радиодетали**.

2.5. Справочник с предопределенными элементами

В заключение мы создадим справочник **Склады**, который будет содержать информацию о складах, используемых ООО «На все руки мастер». Этот справочник будет содержать один предопределенный элемент – склад **Основной**, на который будут поступать все материалы.

53. Создайте справочник: выберите на дереве объектов конфигурации ветвь **Справочники**, **МП**, выберите **Добавить**, на вкладке **Основные** в поле **Имя** введите **Склады**, нажмите **tab** и в поле **Синоним** должно появиться **Склады**, в поле **Представление объекта** введите **Склад**, выберите **Далее**.

54. На вкладке подсистемы выберите **Учет материалов**, **Оказание услуг**. Так как поступление материалов и оказание услуг учитывается в разрезе складов.

55. Выберите вкладку **Формы**, установите флажок **Быстрый просмотр**.

Свойство **Быстрый выбор** позволяет выбирать элементы не из отдельной формы, а из небольшого выпадающего списка,

заполненного элементами этого справочника. Это вариант наиболее удобен для списка складов, так как их будет немного.

56. Выберите вкладку **Прочее, Предопределенные**, система откроет список предопределенных элементов справочника.

57. Сейчас он пуст, поэтому выберите **Добавить**, в поле **Имя** введите **Основной**, в поле **Наименование** — **Основной**, нажмите **ОК**.

Используя встроенный язык мы можем обратиться к этому элементу, используя имя, присвоенное ему в конфигураторе. Наименование предопределенного элемента справочника пользователь может изменить, а имя пользователь не видит и изменить не может.

58. Сделайте доступной в панели действий раздела **Учет материалов** стандартную команду для создания новых складов: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы, МП, Все подсистемы**, в окне **Все подсистемы** слева в списке **Подсистемы** выберите подсистему **УчетМатериалов**, в группе **Панель действий.Создать** для команды **Склад.создать** включите флажком видимость.

59. Проверим наши изменения в режиме **1С:Предприятие**: выберите **Отладка, Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне **Реорганизация информации** выберите **Принять**.

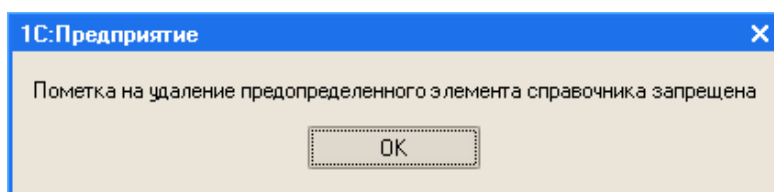
60. Создайте элементы справочника: выберите раздел **Учет материалов**, в панели навигации выберите **Склады**, введите новый элемент склад **Розничный**.

61. Создайте копию информационной базы: выберите **Администрирование, Выгрузить информационную базу**, в поле **Имя файла** введите **Посад_2**, нажмите **Сохранить**, после появления сообщения **Выгрузка информационной базы в файл завершена**, нажмите **ОК**.

2.6. Предопределенные элементы

Обратите внимание, что система отмечает различными пиктограммами простой и предопределенный элементы справочника. Несмотря на то что можно изменить код или наименование у обоих элементов, пометка на удаление (или удаление) возможна только для простых

элементов справочника. При попытке пометить на удаление predetermined элемент система выдаст предупреждение:



Таким образом, теперь мы можем обозначить две характерные особенности predetermined элементов:

- на predetermined элементы могут опираться алгоритмы работы конфигурации (т.к. возможно обращение к ним из встроенного языка по имени),
- predetermined элементы являются объектами базы данных, которые нельзя удалить в режиме 1С:Предприятия.

Из этого видно, в чем заключается принципиальная с точки зрения конфигурации разница между обычными и predetermined элементами справочника.

Обычные элементы «непостоянны» для конфигурации. В процессе работы пользователя они могут появиться, исчезнуть. Поэтому конфигурация хоть и может отличить их друг от друга, но рассчитывать на них в выполнении каких-либо алгоритмов она не может в силу их «непостоянства».

Predetermined элементы, напротив, «постоянны». В процессе работы пользователя они находятся всегда на своих местах и исчезнуть не могут. Поэтому с ними конфигурация может работать вполне уверенно и опираться на них при отработке различных алгоритмов. По этой причине каждый из predetermined элементов имеет уникальное имя для того, чтобы к нему можно было обратиться средствами встроенного языка.

2.7. Основная конфигурация и конфигурация базы данных

До сих пор мы не углублялись в структуру системы 1С:Предприятие 8. Помните: с точки зрения пользователя программа 1С состоит из платформы и конфигурации. Мы говорили, что в каждом конкретном случае используется одна из множества возможных конфигураций. Настало время сказать, что это не совсем так.

Почему *не* так? Потому что в каждой информационной базе существуют как минимум две конфигурации.

Почему не *совсем* так? Потому что пользователь действительно работает всегда только с одной конфигурацией. Вторая конфигурация предназначена для разработчика или человека, который должен вносить изменения в конфигурацию (например, администратора базы данных). Для пользователя она «не видна».

Конфигурация, предназначенная для разработчика, называется *Основная конфигурация* (или просто *Конфигурация* – та, которую мы редактировали в Конфигураторе). Конфигурация, с которой работают пользователи, называется *Конфигурация базы данных*.

Основную конфигурацию можно редактировать. Конфигурацию базы данных редактировать нельзя, можно только произвести обновление конфигурации базы данных на основе основной конфигурации.

Такое внутреннее устройство позволяет вносить изменения в конфигурацию, не прерывая работы пользователей (поскольку изменения вносятся в основную конфигурацию). Затем, когда разработчик будет уверен в том, что все изменения, которые он внес, верны, можно будет быстро произвести обновление конфигурации базы данных, используя основную конфигурацию. Но для этого придется завершить работу всех пользователей.

Разработчик всегда может сравнить основную конфигурацию и конфигурацию базы данных, может вернуться к исходному состоянию основной конфигурации, используя конфигурацию базы данных (если, например, совсем запутался в своих изменениях).

Таким образом, взаимодействие двух конфигураций можно представить следующим образом (рис. 2.24):



Рис. 2.24. Взаимодействие двух конфигураций

Когда разработчик работает с основной конфигурацией, система всегда подсказывает ему, отличается ли его вариант основной конфигурации от того, который сохранен, и отличается ли сохраненный вариант основной конфигурации от конфигурации базы данных.

Если разработчик редактирует основную конфигурацию и редактируемый вариант основной конфигурации отличается от того, кото-

рый сохранен, в заголовке окна дерева конфигурации появляется признак модифицированности конфигурации (*) – рис. 2.25:



Рис. 2.25. Заголовок окна дерева конфигурации

Если сохраненный вариант основной конфигурации отличается от конфигурации базы данных, в заголовке окна дерева конфигурации появляется знак отличия конфигураций (<!>) – рис. 2.26:



Рис. 2.26. Заголовок окна дерева конфигурации

Для сохранения основной конфигурации следует воспользоваться командой **Конфигурация** ▶ **Сохранить конфигурацию**, а для обновления конфигурации базы данных необходимо выполнить команду **Конфигурация** ▶ **Обновить конфигурацию базы данных**. При выполнении команды **Отладка** ▶ **Начать отладку** система сама сначала сохраняет основную конфигурацию, а затем производит ее сравнение с конфигурацией базы данных. В случае если конфигурации отличаются, выдается запрос на обновление конфигурации базы данных, который вы видели в предыдущих примерах.

При выполнении команды **Отладка** ▶ **Продолжить** система, после описанных выше действий, предлагает еще и перезапустить приложение, чтобы прекратить текущую отладочную сессию.

Таким образом, система старается облегчить жизнь разработчика и автоматизировать часто выполняемые операции.

Важным фактом является то, что именно в момент обновления конфигурации базы данных система создает (модифицирует) в базе данных те структуры хранения данных, которые мы описали в виде объектов конфигурации.

Таким образом, обычные элементы справочника пользователь добавляет в ту структуру базы данных, которую создала система на основе объекта конфигурации **Справочник**, а predetermined элементы этого справочника система добавляет в эту структуру сама, на основе все того же описания этой структуры, которым является объект конфигурации **Справочник**.

Отсюда следует немаловажный факт (о котором говорилось в предыдущем разделе), что если простые элементы справочника «безразличны» для конфигурации, то predetermined элементы важны для нее, поскольку на них могут быть «завязаны» алгоритмы работы конфигурации.

2.8. Палитра свойств

Еще один инструмент разработчика, который мы использовали по ходу изложения, но на описании которого не заостряли внимание, – это **палитра свойств**.

Если помните, мы использовали палитру свойств, когда создавали табличную часть и реквизиты табличной части объекта конфигурации **Справочник** **Номенклатура**.

Палитра свойств – это специальное служебное окно, которое позволяет редактировать все существующие свойства объекта конфигурации. Поскольку разные объекты конфигурации имеют самые разные свойства, содержимое этого окна будет меняться в зависимо-

сти от того, какой объект является текущим (на каком объекте конфигурации установлен курсор).

При некоторых действиях разработчика (например, создание табличной части) палитра свойств открывается автоматически. Но разработчик всегда может открыть палитру свойств объекта конфигурации самостоятельно, воспользовавшись пунктом **Свойства** контекстного меню правой кнопки мыши.

В этом случае палитра свойств откроется и будет закреплена на рабочей области конфигуратора. Однако есть удобная возможность «открепить» палитру свойств (используя символ кнопки в заголовке окна палитры свойств – рис. 2.27):

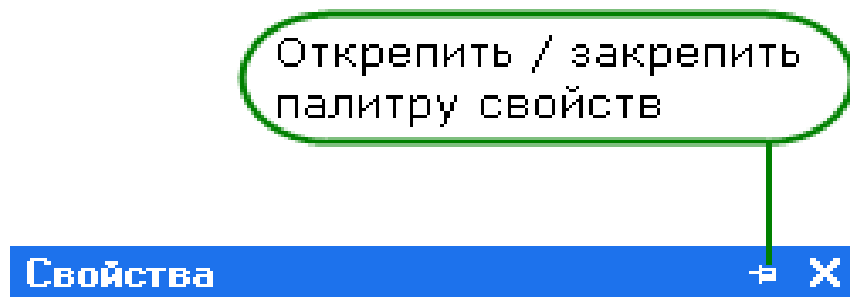


Рис. 2.27. «Открепим» палитру свойств...

В этом состоянии, при наведении курсора мыши на любое другое окно, палитра свойств будет сворачиваться на дополнительную панель в правой части экрана (по умолчанию) (рис. 2.28):

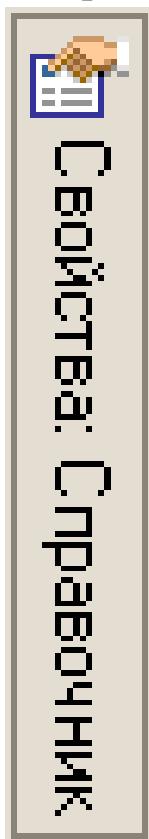


Рис. 2.28. Кнопка на дополнительной панели

А при наведении курсора мыши на символ свернутой палитры свойств она будет открываться.

Подобным поведением (возможностью быть прикрепленным, прячущимся и т.д.) обладает не только окно палитры свойств, но и другие окна конфигуратора (например, окно дерева конфигурации).

2.9. Контрольные вопросы

- Для чего предназначен объект конфигурации Справочник.
- Каковы характерные особенности справочника.
- Для чего используются реквизиты и табличные части справочника.
- Зачем нужны иерархические справочники и что такое родитель.
- Зачем нужны подчиненные справочники и что такое владелец.
- Какие основные формы существуют у справочника.
- Что такое предопределенные элементы.
- Чем с точки зрения конфигурации отличаются обычные элементы справочника от предопределенных элементов.
- Как пользователь может отличить обычные элементы справочника от предопределенных элементов.
- Как создать объект конфигурации Справочник и описать его структуру.
- Как добавить новые элементы в справочнике.
- Как создать группу справочника.
- Как переместить элементы из одной группы справочника в другую.
- Зачем нужна основная конфигурация и конфигурация базы данных.
- Как изменить конфигурацию базы данных.
- Как связаны объекты конфигурации и объекты базы данных.
- Что такое подчиненные объекты конфигурации.
- Зачем нужна проверка заполнения у реквизитов справочника.
- Что такое быстрый выбор и когда его использовать.
- Как отобразить справочник и определить его представление в различных разделах интерфейса приложения.
- Как отобразить команды создания нового элемента справочника в интерфейсе подсистем.

- Как редактировать командный интерфейс подсистем.

3-й день. Документы

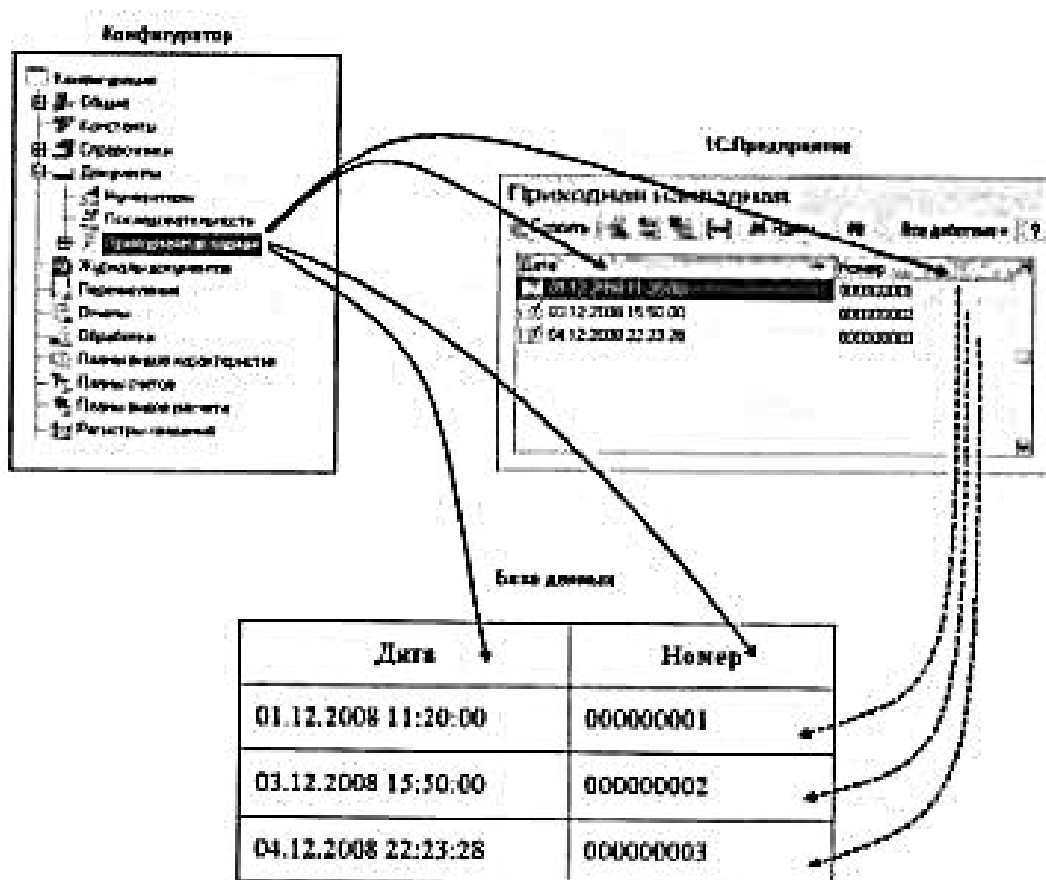
Носить тяжести — вот призвание мужчины... Не ослабевая, мужчина выдерживал эту судьбу носильщика на протяжении веков. Хотя он и выдрессировал животных, чтобы те помогали ему в этом занятии, из игры он не вышел, а роль свою сохранил.

Г. Башляр. Психология тяжести и тяготения

3.1. Что такое документ

Документы - это прикладные объекты конфигурации. Они позволяют хранить в прикладном решении информацию о совершенных хозяйственных операциях или о событиях, произошедших в "жизни" предприятия вообще. Это могут быть, например, приходные накладные, приказы о приеме на работу, счета, платежные поручения и т.д.

В базе данных документ представляет собой отдельную запись в основной таблице, хранящей информацию об этом виде документов.

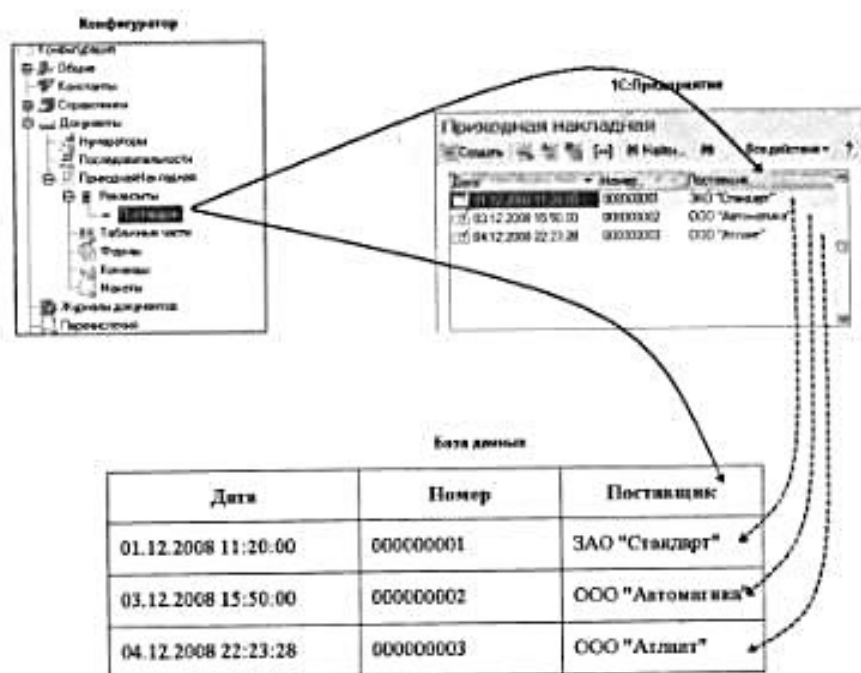


Каждый документ характеризуется номером, датой и временем. Система поддерживает режим автоматической нумерации документов, при котором она самостоятельно может генерировать номер для нового документа. Кроме этого система позволяет осуществлять контроль уникальности номеров документов, не разрешая создавать документы с одинаковыми номерами.

Система автоматически поддерживает режим, при котором уникальность номеров и автоматическая нумерация могут выполняться в пределах определенного периода (день, месяц, квартал, год). Например, если периодичность установлена год, то с нового года система опять начнет нумерацию указанных документов с 1.

Важными характеристиками документа являются дата и время. Они позволяют установить строгую временную последовательность совершения операций. Таким образом, документы могут отличаться друг от друга не только номером, но и своим положением на временной оси. В результате всегда можно сказать, какая из двух операций была совершена раньше.

Помимо номера, даты и времени, каждый документ, как правило, содержит некоторую дополнительную информацию, которая подробно описывает этот документ. Например, для документа **Поступление товаров и услуг** это может быть информация о поставщике товаров (контрагенте), складе, на который приходятся товары и т.п. Набор такой информации является одинаковым для всех документов конкретного вида, и для ее хранения служат реквизиты документа.



Кроме этого, каждый документ может содержать некоторый набор информации, которая одинакова по своей структуре, но различна по количеству, для разных документов. Например, для документа **Поступление товаров и услуг** это может быть информация о товарах, поступивших на предприятие (наименование, количество, и т.д.), серийных номерах и возвратной таре. Для хранения подобных данных служат табличные части документа.



Важным свойством документа является возможность его проведения. Если документ проводится, то он может изменить состояние тех или иных учитываемых данных. Если же документ не является "проводимым" это значит, что событие, которое он отражает, не влияет на состояние учета, который ведется в данном прикладном решении.

Например, документ **Поступление товаров и услуг** при своем проведении может вносить изменения в состояние расчетов с поставщиками, в учет остатков товаров, в состояние заказов покупателей и другие учетные данные

3.2. Документ Приходная накладная

62. Загрузите информационную базу: выберите **Администрирование**, **Загрузить информационную базу**, выберите **Посад_2**, нажмите **Открыть**, после появления сообщения **После загрузки информационной базы работа Конфигуратора будет завершена. Несохранные данные в открытых окнах могут быть потеряны! Продолжить?** Нажмите **Да**. После сообщения **Информационная база успешно загружена. Работа Конфигуратора будет завершена. Перезапустить Конфигуратор?** Нажмите **Да**.

К услугам выполняемым нашим предприятием относятся ремонт телевизоров и установка стиральных машин. Для этого требуются некоторые материалы, которые расходуются в процессе оказания этих услуг. Поэтому важными событиями в жизни нашей организации будут поступление материалов и оказание услуг.

Создадим документ **Приходная накладная**, который будет фиксировать факт поступления в организацию необходимых материалов, и документ **Оказание услуг** – фиксировать оказание услуг и расход материалов, используемых при оказании этих услуг.

63. Создайте документ: выберите на дереве объектов конфигурации ветвь **Документы**, нажмите **МП**, выберите **Добавить**, на вкладке **Основные** в поле **Имя** введите **Приходная Накладная**, нажмите **tab** и в поле **Синоним** должно появиться **Приходная Накладная**, представление создавать не будем, вместо него будем использовать **Синоним объекта**, так как мы создали имя в единственном числе, в поле **Представление списка** введите **Приходные накладные**, выберите **Далее**.

64. На вкладке подсистемы выберите **Учет материалов** и **Бухгалтерия**, так при к учету материалов этот документ имеет прямое отношение, а для бухгалтерского анализа всегда может понадобиться список документов, отражающих поступление материалов.

65. Выберите вкладку **Данные**, выберите **Добавить** над списком реквизитов документа, в поле **Имя** введите **Склад**, в списке **Тип** выберите **Справочник Ссылка.Склады**.

66. В палитре свойств реквизита выберите **Значение заполнения**, выберите **предопределенный элемент справочника Склады – Основной**.

67. Создайте новую табличную часть: нажмите **Добавить табличную часть** над списком табличных частей документа, в поле **Имя** введите **Материалы**, в свойстве **Проверка заполнения** выберите значение **Выдавать ошибку**.

Тем самым задаем условие, что документ обязательно должен содержать табличную часть, то есть список приходуемых материалов.

68. Создайте реквизит табличной части: выберите табличную часть **Материалы**, **МП**, **Добавить реквизит** в разделе описания табличных частей документа, в поле **Имя** введите **Материал**, в списке **Тип** выберите **СправочникСсылка.Номенклатура**.

69. Создайте реквизиты табличной части:

Количество, тип **Число**, длина **15**, точность **3**, неотрицательное.

Цена, тип **Число**, длина **15**, точность **2**, неотрицательное,

Сумма, тип **Число**, длина **15**, точность **2**, неотрицательное.

70. Для каждого реквизита табличной части в свойстве **Проверка заполнения** выберите значение **Выдавать ошибку**.

71. Выберите вкладку **Нумерация**, свойство **Автонумерация** должно быть включено. Это обеспечит автоматическую генерацию уникальных номеров для создаваемых нами документов.

72. Сделайте доступной в панели действий раздела **Учет материалов** стандартную команду для создания новых материалов: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы**, **МП**, **Все подсистемы**, в окне **Все подсистемы** слева в списке **Подсистемы** выберите подсистему **УчетМатериалов**, в группе **Панель действий.Создать** для команды **Приходная накладная.создать** включите **видимость**.

73. Проверим наши изменения в режиме 1С:Предприятие: выберите **Отладка**, **Начать отладку**, на вопрос **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне **Реорганизация информации** выберите и.

74. Создайте новую приходную накладную: выберите раздел **Учет материалов**, в панели действий выберите **Приходная накладная**.

Система автоматически подставляет текущую дату создания документа и нулевое время, так как документ еще не проведен. В

качестве времени документа при оперативном проведении ему присваивается оперативная отметка времени.

Поле Номер не заполнено, но система сама генерирует для нового документа уникальный номер, так как свойство Автонумерация для документа включено по умолчанию.

Поле Склад уже заполнено значением Основной, так это задано в свойствах этого реквизита.

75. Заполните табличную часть: нажмите **Insert**, заполним ее материалами для ремонта телевизоров: **Строчный трансформатор GoldStar, Количество – 10, Цена – 270, Сумма 2700, Строчный трансформатор Samsung, Количество – 10, Цена – 600, Сумма – 6000, Транзистор Philips, Количество – 10, Цена – 3, Сумма – 30,** нажмите **Провести и закрыть**.

76. Создайте и проведите документ, который будет приходовать следующие материалы для установки стиральных машин **Кабель электрический, Количество – 5, Цена – 20, Сумма – 100, Шланг резиновый, Количество – 5, Цена – 100, Сумма – 500.**

При вводе материалов не используйте кнопку выбора три точки, а просто вводите название материалов в это поле. Платформа автоматически найдет материалы, наименование которых начинается с введенных символов, и предложит их для выбора.

3.3. Автоматический пересчет суммы в строках документа

При заполнении документа приходится вводить сумму в каждой строке. Это неудобно, и возникает желание автоматизировать работу документа так, чтобы сумма вычислялась автоматически каждый раз при изменении цены или количества материалов в строке.

Теперь надо слегка изменить логику работы формы документа, а значит, нам придется создать свою собственную форму документа «Приходная Накладная» для того, чтобы в ней мы могли описать тот алгоритм, который нам нужен.

77. Создайте форму документа: в окне Конфигуратора выберите на дереве объект **Документы**, выберите **Приходная Накладная, М2**, выберите вкладку **Формы**, можно видеть, что ни одна из основных форм документа не задана, выберите кнопку **Лупа** в поле **Документа** или нажмите кнопку **Добавить** над списком форм, должен запускаться конструктор форм, выберите **Форма документа**, и огласившись со всем, что нам предлагает система нажмите **Готово**.

В дереве объектов конфигурации у объекта конфигурации Документ Приходная Накладная появилась форма Форма Документа, а на экране открылось окно редактора форм, содержащее эту форму.

Форма документа «Приходная Накладная» содержит большое количество всевозможных полей. Эти поля называются элементами управления. Они имеют разное назначение и разное поведение, которое соответствует их назначению. Однако все они служат для того, чтобы отображать информацию, хранящуюся в базе данных и организовывать интерактивную работу с этой информацией.

При разработке форм объектов конфигурации разработчик не имеет возможности нарисовать форму. Он может только указать из каких элементов будет состоять форма, а система уже сама самостоятельно расположить эти элементы в форме.

Элементы в форме в **верхнем левом** окне редактора форм образуют иерархическую структуру, из которой следует, что чем выше в списке находится элемент, тем выше и левее на форме он будет располагаться.

Эта структура редактируется на вкладке Элементы и позволяет управлять отображением и редактированием данных в форме.

Мы хотим, чтобы каждый раз, когда меняется значение в поле «Количество» или в поле «Цена», в поле «Сумма» автоматически устанавливалось значение равное $\text{Количество} * \text{Цена}$. Очевидно, что для этого нужно написать на встроенном языке команду похожую на $\text{Сумма} = \text{Количество} * \text{Цена}$, которая будет выполняться при изменении значения поля «Количество» или «Цена».

3.4. Обработчик события

У системы существуют события, которые связанные с самыми различными моментами ее стандартного поведения. Используя встроенный язык разработчик может вклиниться в эти события и описать собственный алгоритм того, что должно происходить при наступлении этого события.

78. Выберите элемент формы **МатериалыКоличество**, **МП**, **Свойства**, должна появиться Палитра свойств, прокрутите список до конца, вы обнаружите перечень событий, которые могут быть связаны с этим полем ввода, в группе **События** в поле **При изменении** нажмите **Лупа**, система создаст заготовку процедуры обработчика этого события в модуле нашей формы.

Модуль — это «хранилище» для текста программы на встроенном языке. В конфигурации существует большое количество модулей, которые расположены в различных ее точках. Они могут принадлежать некоторым объектам конфигурации (например, формам), а могут существовать сами по себе (принадлежать всей конфигурации в целом).

79. В модуль формы добавьте код:

```
&НаКлиенте
Процедура МатериалыКоличествоПриИзменении(Элемент)
СтрокаТабличнойЧасти=Элементы.Материалы.ТекущиеДанные;
СтрокаТабличнойЧасти.Сумма=СтрокаТабличнойЧасти.Количество*СтрокаТабличнойЧасти.Цена;
КонецПроцедуры
```

В первой строке мы сначала создаем переменную `СтрокаТабличнойЧасти`, в которую будет помещен объект, содержащий данные, находящиеся в строке табличной части, которую нам нужно пересчитать.

Мы создаем переменную прямо по ходу работы, и ее тип определяется типом значений, которые она содержит.

Поскольку мы находимся в модуле формы, то в нем доступны все свойства и методы объекта встроенного языка `УправляемаяФорма`. Поэтому можем обращаться к ней напрямую. После знака равенства мы обращаемся к коллекции элементов формы, используя одно из свойств объекта `УправляемаяФорма` – свойство `Элементы`.

Коллекция элементов формы является объектом встроенного языка `ВсеЭлементыФормы.`, содержащее все элементы формы.

Каждый элемент формы можно получить, указав его имя в качестве свойства этого объекта, то есть через точку от него. В данном случае мы обращаемся к табличной части документа «Материалы» (`Элементы.Материалы`).

Табличная часть документа представляет собой объект встроенного языка `ТаблицаФормы`. Получить ту строку, в которой в настоящее время осуществляется редактирование, можно при помощи свойства программного объекта `ТаблицаФормы` — `ТекущиеДанные` (`Элементы.Материалы.ТекущиеДанные`).

Таким образом, в результате выполнения первой строки переменная СтрокаТабличнойЧасти будет содержать объект ДанныеФормыСтруктура. Этот объект содержит данные, находящиеся в текущей строке табличной части документа (Элементы.Материалы.ТекущиеДанные).

Получив этот объект, мы можем обратиться к данным конкретной колонки табличной части, указав имя колонки в качестве свойства объекта. Например, используя обращение СтрокаТабличнойЧасти.Количество мы получаем число, которое находится в редактируемой строке в колонке Количество.

То есть во второй строке процедуры обработчика вычисляется значение колонки Сумма как произведение значений колонок Количества и Цена.

80. Выберите **Отладка, Начать отладку**, на вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

81. Теперь посмотрим, как это работает: выберите любой документ **Приходная Накладная**, теперь поменяйте количество в любой строке документа, сумма в строке должна будет пересчитана.

Замечательно. Но теперь хотелось бы и для поля «Цена» сделать то же самое. А если заглянуть вперед, то мы увидим, что подобное автоматическое заполнение поля «Сумма» может нам понадобиться и в других документах. Поэтому лучше будет поместить расчет суммы в некотором «общедоступном» месте, чтобы разные документы, имеющие аналогичные реквизиты табличной части, могли использовать этот алгоритм.

3.5. Одна процедура для обработки нескольких события

82. Создадим объект конфигурации Общий модуль: в дереве объектов конфигурации выберите ветвь **Общие, Общие модули, МП, Добавить**, в поле **Имя** введите **РаботаСДокументами**, выберите в свойствах **Клиент** (управляемое приложение), а флажок **Сервер** отключите.

83. Выберите **Модуль, Открыть**, введите код:

**Процедура РассчитатьСумму(СтрокаТабличнойЧасти)
Экспорт**

СтрокаТабличнойЧасти.Сумма=СтрокаТабличнойЧасти.Количество * СтрокаТабличнойЧасти.Цена;

КонецПроцедуры

84. Затем в модуле нашей формы изменим текст нашего обработчика: выберите **Документы, ПриходнаяНакладная**, выберите **Формы, ФормаДокумента, М2**, выберите **Модуль**, измените код так:

&НаКлиенте

Процедура МатериалыКоличествоПриИзменении(Элемент)

СтрокаТабличнойЧасти=Элементы.Материалы.ТекущиеДанные;

РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);

КонецПроцедуры

85. Теперь осталось и для поля «Цена» установить такой же обработчик. Поэтому мы создадим обработчик события «При изменении» для поля ввода, которое расположено в колонке «Цена» и повторим в нем вызов процедуры «РассчитатьСумму» из общего модуля: выберите **Приходная накладная, Форма, ФормаДокумента, М2**, выберите элемент формы **МатериалыЦена, МП, Свойства**, должна появиться Палитра свойств, прокрутите список до конца, вы обнаружите перечень событий, которые могут быть связаны с этим полем ввода, в группе **События** в поле **При изменении** нажмите **Лупа**, система создаст заготовку процедуры обработчика этого события в модуле нашей формы.

86. Введите код:

&НаКлиенте

Процедура МатериалыЦенаПриИзменении(Элемент)

СтрокаТабличнойЧасти=Элементы.Материалы.ТекущиеДанные;

РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);

КонецПроцедуры

87. Теперь посмотрим, как это работает: выберите любой документ **ПриходнаяНакладная**, поменяйте количество в любой строке документа, сумма в строке должна будет пересчитана, теперь поменяйте цена в любой строке документа, сумма в строке должна будет пересчитана.

3.6. Документ Оказание услуги

88. Создайте документ: выберите на дереве объектов конфигурации ветвь **Документы**, нажмите **МП**, выберите **Добавить**, на вкладке **Основные** в поле **Имя** введите **ОказаниеУслуги**, нажмите **tab** и в поле **Синоним** должно появиться **Оказание услуги**, представление создавать не будем, вместо него будем использовать Синоним объекта, так как мы создали имя в единственном числе, в поле **Представление списка** введите **Оказание услуг**, выберите **Далее**.

89. На вкладке подсистемы выберите **Оказание услуг** и **Бухгалтерия**, так при оказанию услуг этот документ имеет прямое отношение, а для бухгалтерского анализа всегда может понадобится список документов, отражающих оказание услуг.

90. Выберите вкладку **Данные**, выберите **Добавить** над списком реквизитов документа, в поле **Имя** введите **Склад**, в списке **Тип** выберите **СправочникСсылка.Склады**.

91. В палитре свойств реквизита выберите **Значение заполнения**, выберите предопределенный элемент справочника **Склады – Основной**.

92. Выберите вкладку **Данные**, выберите **Добавить** над списком реквизитов документа, в поле **Имя** введите **Клиент**, в списке **Тип** выберите **СправочникСсылка.Клиенты**.

93. Выберите вкладку **Данные**, выберите **Добавить** над списком реквизитов документа, в поле **Имя** введите **Мастер**, в списке **Тип** выберите **СправочникСсылка.Сотрудники**.

94. Для каждого реквизита в свойстве **Проверка заполнения** выберите значение **Выдавать ошибку**.

95. Создайте новую табличную часть: нажмите **Добавить табличную часть** над списком табличных частей документа, в поле **Имя** введите **ПереченьНоменклатуры**, в свойстве **Проверка заполнения** выберите значение **Выдавать ошибку**.

96. Создайте реквизит табличной части: выберите табличную часть **ПереченьНоменклатуры**, **МП**, **Добавить реквизит** в разделе описания табличных частей документа, в поле **Имя** введите **Номенклатура**, в списке **Тип** выберите **СправочникСсылка.Номенклатура**.

97. Создайте реквизит табличной части:

Количество, тип – **Число**, длина – **15**, точность – **3**, неотрицательное.

Цена, тип – **Число**, длина – **15**, точность – **2**, неотрицательное,

Сумма, тип – **Число**, длина – **15**, точность – **2**, неотрицательное.

98. Для каждого реквизита в свойстве **Проверка заполнения** выберите значение **Выдавать ошибку**.

99. Выберите вкладку **Нумерация**, свойство **Автонумерация** должно быть **включено**. Это обеспечит автоматическую генерацию уникальных номеров для создаваемых нами документов.

100. Создайте форму документа: выберите вкладку **Формы**, можно выберите кнопку **Луна** в поле **Формы Документа** или нажмите кнопку **Добавить** над списком форм, должен запуститься конструктор форм, выберите **Форма документа**, нажмите **Готово**.

101. Выберите элемент формы **ПереченьНоменклатурыКоличество**, **МП**, **Свойства**, должна появиться **Палитра свойств**, прокрутите список до конца, вы обнаружите перечень событий, которые могут быть связаны с этим полем ввода, в группе **События** в поле **При изменении** нажмите **Луна**, система создаст заготовку процедуры обработчика этого события в модуле нашей формы.

102. Выберите элемент формы **ПереченьНоменклатурыЦена**, **МП**, **Свойства**, должна появиться **Палитра свойств**, прокрутите список до конца, вы обнаружите перечень событий, которые могут быть связаны с этим полем ввода, в группе **События** в поле **При изменении** нажмите **Луна**, система создаст заготовку процедуры обработчика этого события в модуле нашей формы.

103. Введите код:

&НаКлиенте

Процедура

ПереченьНоменклатурыКоличествоПриИзменении(Элемент)

СтрокаТабличнойЧасти=Элементы.ПереченьНоменклатуры.ТекущиеДанные;

РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);

КонецПроцедуры

&НаКлиенте

Процедура

ПереченьНоменклатурыЦенаПриИзменении(Элемент)

**СтрокаТабличнойЧасти=Элементы.ПереченьНоменклатуры.
ТекущиеДанные;**

**РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧа
сти);**

КонецПроцедуры

104. Сделайте доступной в панели действий раздела Учет материалов стандартную команду для создания новых материалов: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы, МП, Все подсистемы**, в окне Все подсистемы слева в списке Подсистемы выберите подсистему **Оказание услуг**, в группе Панель действий.Создать для команды **Оказание услуги.создать** включите видимость.

105. Проверим наши изменения в режиме 1С:Предприятие: выберите **Отладка, Начать отладку**, на вопрос конфигууратора **Редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

106. Создайте документ: выберите раздел **Оказание услуг**, в панели действий выберите **Оказание услуги**.

107. Заполните табличную часть: нажмите **Insert**, заполним ее **Транзистор Philips, Количество – 1, Цена – 3, Сумма – 3**, нажмите **Провести и закрыть**.

3.7. Анализ кода с помощью синтакс-помощника

Пользоваться синтакс-помощником удобно в тех случаях, когда нужно разобраться в уже написанном незнакомом коде.

108. Откройте программный модуль: выберите **Приходная накладная, M2, Форма, ФормаДокумента, M2**, выберите **Модуль**, откройте текст процедуры **МатериалыКоличествоПриИзменении**.

109. Используем контекстную помощь синтакс-помощника: выберите курсором на выражении **Элементы.Материалы.ТекущиеДанные** и нажмите **Ctrl+F1**.

Синтакс-помощник откроется на закладке **Индекс** и выражение **Элементы** будет помещено в строку поиска. Среди конструкций встроеного языка, отсортированных по алфавиту, будет произведен поиск этого выражения.

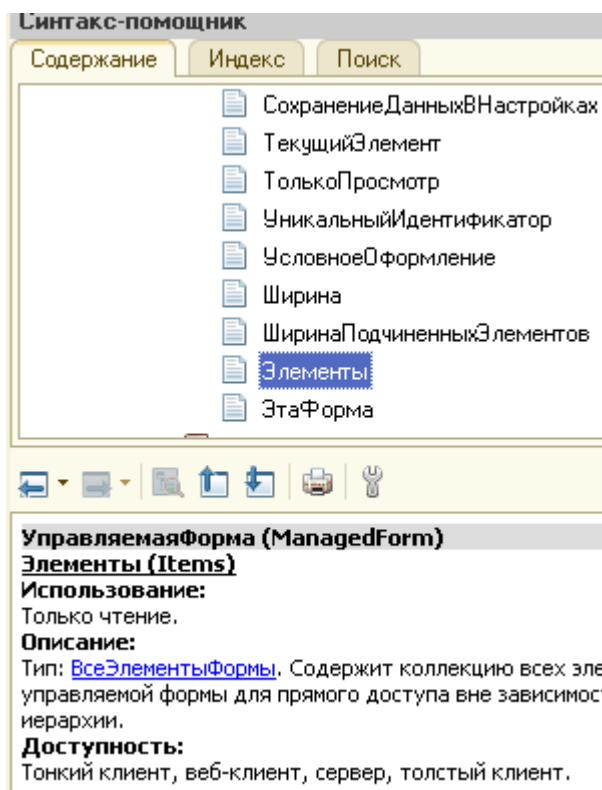
Должно появиться окно со списком глав, в которых это выражение используется.

110. Но вручную просматривать список глав сложно, поэтому находясь в окне этого списка нажмите **Ctrl+F**, в окне Поиск в поле Искать введите **УправляемаяФорма**, нажмите **Искать**. Будет выделена нужная глава.

111. Просмотрите главу: нажмите **Показать**.

112. Чтобы найти, в какой ветки дерева находится открытое описание нажмите кнопку **Найти текущий элемент в дереве**. Таким образом в дереве содержания мы увидим, что **Элементы** – это свойство объекта **УправляемаяФорма**.

113. В описании **Элементы** следует, что, используя свойство **Элементы**, мы получаем объект **ВсеЭлементыФормы**, который содержит коллекцию всех элементов формы.



114. Нажмите ссылку Тип: **ВсеЭлементыФормы**.

ВсеЭлементыФормы (FormAllItems)
Элементы коллекции:
[ГруппаФормы](#), [ТаблицаФормы](#), [ПолеФормы](#), [КнопкаФормы](#)
 Для объекта доступен обход коллекции посредством оператора Для каждого ... Из ... Цикл. При обходе выбираются элементы. Возможно обращение к элементу посредством оператора [...]. В качестве аргумента передается имя.
Свойства:
[<Имя элемента управления> \(<Control name>\)](#)
Методы:
[Вставить \(Insert\)](#)
[Добавить \(Add\)](#)
[Количество \(Count\)](#)
[Найти \(Find\)](#)
[Переместить \(Move\)](#)
[Удалить \(Delete\)](#)
Описание:
 Коллекция всех элементов управляемой формы не зависит от их места в иерархии. В коллекцию не входит только корневой элемент.

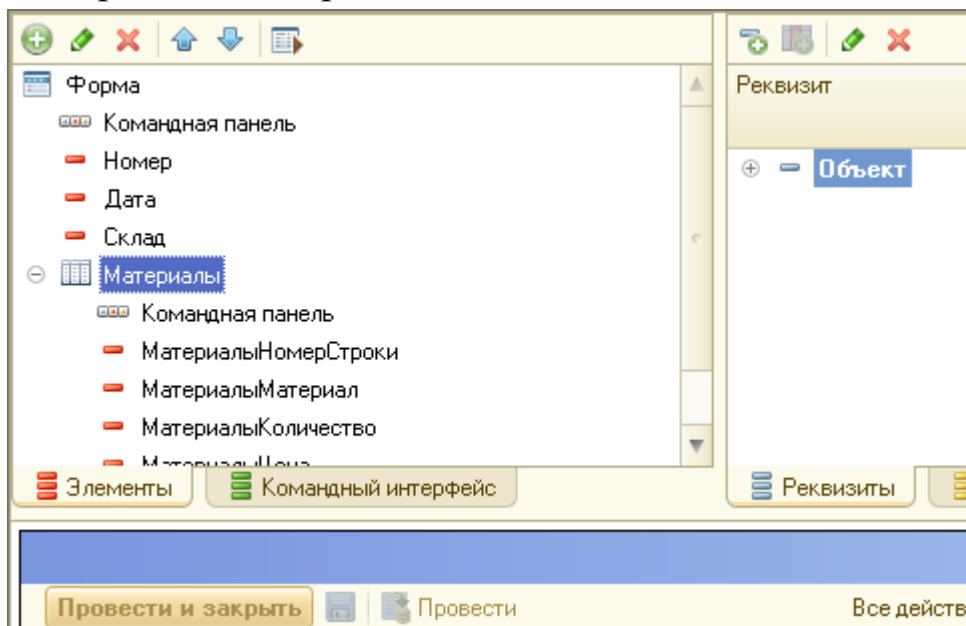
Эта коллекция содержит элементы управляемой формы, размещенные на форме. Доступ к элементу осуществляется по имени.

СтрокаТабличнойЧасти=Элементы.Материалы.ТекущиеДанные;

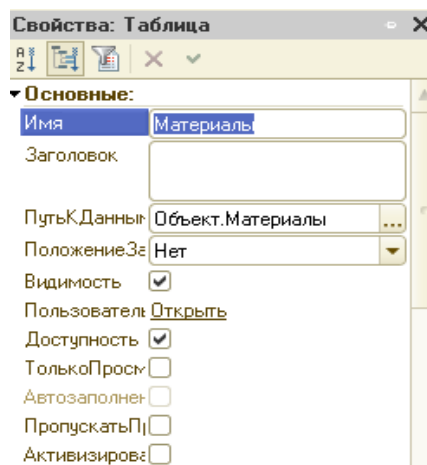
В коллекции есть свойство – <Имя элемента управления>. Ага, значит **Материалы** – имя некоторого элемента формы.

115. Посмотрим структуру элементов формы: откройте форму документа **ПриходнаяНакладная**, выберите вкладку **Форма**, выбо-

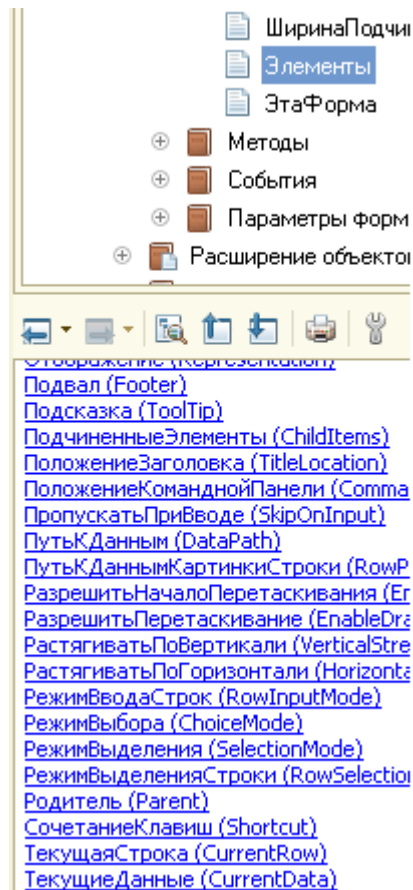
рите вкладку **Элементы**. В структуре элементов формы мы видим таблицу **Материалы** – см. рис.



116. Откройте палитру свойств: выберите **Материалы**, **МП**, **Свойства**. В заголовке мы видим – **Свойства: Таблица**. Значит, этот элемент формы является таблицей и следовательно нам нужен объект коллекции **ВсеЭлементыФормы – ТаблицаФормы**.

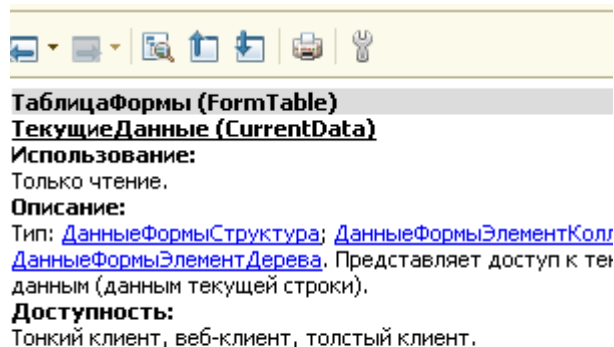


117. В синтакс-помощнике выберите **ТаблицаФормы**. Мы видим список свойств объекта **ТаблицаФормы**.



118. В списке свойств найдите свойство **ТекущиеДанные**. Значит это одно из свойств объекта **ТаблицаФормы**.

119. Выберите ссылку **ТекущиеДанные**. Из его описания следует, что используя свойство **ТекущиеДанные** мы получаем объект **ДанныеФормыСтруктура**, которые содержат данные находящиеся в текущей строке таблицы.



Значит, в результате выполнения строки

СтрокаТабличнойЧасти=Элементы.Материалы.ТекущиеДанные;

в переменной СтрокаТабличнойЧасти окажется объект тип ДанныеФормаСтруктура

Тогда следующая строка обработчика

```
СтрокаТабличнойЧасти.Сумма=  
СтрокаТабличнойЧасти.Количество*  
СтрокаТабличнойЧасти.Цена;
```

содержащая Сумма, Количество и Цена – это какие-то свойства объекта ДанныеФормыСтруктура.

120. Выберите ссылку **ДанныеФормыСтруктура**. Из описания следует, что используя этот объект мы можем обратиться к данным конкретной табличной части, указав имя колонки в качестве свойства объекта. То есть используя выражение **СтрокаТабличнойЧасти.Сумма** мы обращаемся к данным, которые находятся в колонке Сумма текущей таблицы.

```
ДанныеФормыСтруктура (FormDataSet)
Свойства:
<Имя свойства> (<Имя свойства>)
Методы:
Свойство (Property)
Описание:
Структура данных для моделирования данных, редактиру
управляемой форме. Моделирует объект с набором свойств
Для текущих данных таблицы управляемой формы обращение
значениям осуществляется по именам колонок.
Для параметров управляемой формы обращение к значению
осуществляется по именам параметров.
Доступность:
Тонкий клиент, веб-клиент, сервер, толстый клиент.
См. также:
УправляемаяФорма, свойство Параметры
УправляемаяФорма метод РеквизитФормы.Значение
```

3.8. Анализ кода с помощью отладчика

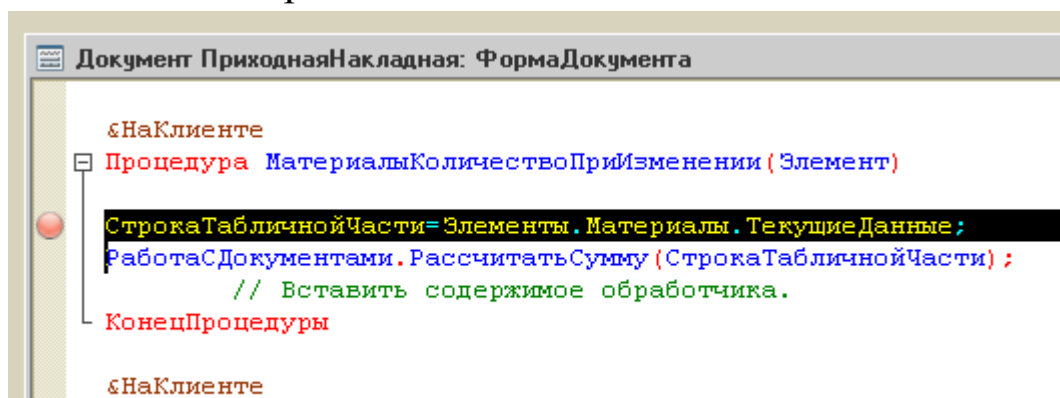
Пользоваться отладчиком удобно в случае написания собственного кода. Можно просто остановиться в конкретном месте программы и посмотреть какие же свойства здесь доступны или какие объекты используются.

121. Откройте программный модуль: выберите **Приходная накладная, М2, Форма, ФормаДокумента, М2**, выберите **Модуль**, откройте текст процедуры **МатериалыКоличествоПриИзменении**.

122. Откройте пункт меню **Отладка**, там стали доступны команды для работы с точками останова.

123. Установите точку останова: выберите в служебной области слева от первой строки процедуры **МатериалыКоличество**

ПриИзменении, М2. В служебной области должен появиться значок точки останова – см. рис.



124. Начнем отладку, запустив систему в отладочном режиме, который мы использовали и раньше, только не устанавливая точек останова и программа не прерывалась: выберите **Отладка, Начать отладку**.

125. Выберите любой документ **ПриходнаяНакладная**, поменяйте количество в любой строке документа, сумма в строке должна будет пересчитана, но выполнение программы **прервется**.

В конфигураторе будет открыта процедура в точке останова. Появится стрелка указывающая на текущую исполняемую строку модуля.

126. Выберите меню **Отладка**, там стали доступны команды для работы с конфигурацией в процессе отладки: **Шагнуть через, Шагнуть в, Шагнуть из** (пошаговое выполнение) или продолжить отладку (**Продолжить отладку**) до следующей точки останова.

Табло и **Вычислить выражение** позволяют получить интересующие нас выражения в каждый момент останова программы.

С помощью **Стек вызовов** можно проследить последовательность вызова процедур и функций.

127. В коде выделите слово **Элементы** и нажмите **Shift+F9**, в поле **Выражение** попало слово **Элементы**. Мы видим, что объект **Элементы** является коллекцией значений **ВсеЭлементыФормы**.

128. Раскройте **Элементы**, нажав на **плюсик**.

129. Выберите **Материалы**. Видим, что это объект **ТаблицаФормы**.

130. Раскройте **Материалы**. Найдите свойство **ТекущиеДанные**. Раскройте его. Вы увидите данные текущей строки табличной части, их данные и тип.

131. Закройте окно **Выражение**.

132. Мы остановились на первой строке процедуры и она еще не выполнялась, поэтому значения переменных еще не заполнены. Чтобы увидеть их после выполнения строки нажмите кнопку **Шагнуть через**. Программа остановится на следующей строке.

133. Нажмите **Шагнуть в**, так как надо шагнуть в процедуру общего модуля **РаботаСДокументами**, где вычисляется значение переменной **СтрокаТабличнойЧасти**. Программа перейдет в процедуру **РассчитатьСумму** общего модуля **РаботаСДокументами**.

134. Выделите **СтрокаТабличнойЧасти**, нажмите **Вычислить выражение**, раскройте объект **СтрокаТабличнойЧасти**.

Теперь переменная **СтрокаТабличнойЧасти** содержит объект **ДанныеФормы.ЭлементКоллекции**. Но значение колонки **Сумма** еще не пересчитаны, так как вторая строка кода еще не исполнялась.

135. Закройте окно **Выражение**.

136. Нажмите **Шагнуть через**. Программа выполнит процедуру **РассчитатьСумму** и остановится на ее конце.

137. Подведите курсор к колонке **Количество** или **Сумма** и система в подсказке покажет текущее значение.

138. Нажмите **Отладка, Завершить**.

Прием

Находясь в модуле формы и при необходимости написать обработчик, можно использовать свойство **ЭтаФорма**, чтобы посмотреть свойства контекста этой конкретной формы.

139. Выберите **Отладка, Начать отладку**.

140. Выберите любой документ **ПриходнаяНакладная**, поменяйте количество в любой строке документа, сумма в строке должна будет пересчитана, но выполнение программы **прервется**.

141. Нажмите **Shift+F9**, в поле **Выражение** введите **ЭтаФорма** и нажмите **Рассчитать**. Раскройте объект и там будет все: типы и свойства объектов встроенного языка, которые используются в момент останова.

142. Нажмите **Отладка, Завершить**.

4-й день. Регистры накопления

Кто же не слышал криков — криков отчаяния или ярости закаленной стали, скрежещущих звуков горячего железа, на которое нападают глубинные воды?.. Так на чьей вы стороне — на стороне огня или воды, мужского или женского начала?

Г. Башляр. Лирический динамизм кузнеца

4.1. Зачем нужен регистр накопления

Итак, мы с вами подошли к одному из главных моментов разработки любой конфигурации – созданию механизма учета накопления данных.

Казалось бы, все необходимое мы с вами уже создали: у нас есть что расходовать и приходовать (справочники), и у нас есть чем расходовать и приходовать (документы). Осталось только построить несколько отчетов, и автоматизация ООО «На все руки мастер» будет закончена.

Однако это не так.

Во-первых, путем анализа документов можно, конечно, получить требуемые нам выходные данные, но представьте, что завтра ООО «На все руки мастер» решит немного изменить свои бизнес-процессы, и нам потребуется ввести в конфигурацию еще один документ (или несколько документов!).

Например, сейчас мы полагаем, что товары поступают в ООО и затем расходуются. Руководство захотело усилить материальный контроль и решило приходовать товары на основной склад организации и затем выдавать их материально ответственным лицам. В этом случае нам придется добавить в конфигурацию еще один документ, который будет фиксировать перемещение материалов между основным складом и материально ответственными лицами. И очевидно, нам придется переработать все отчеты, которые были нами созданы к этому моменту с тем, чтобы они учитывали изменения, вносимые новым документом. А представьте, если в нашей конфигурации не два, а двадцать документов?!

Во-вторых, отчеты, анализирующие документы, будут работать довольно медленно, что будет вызывать раздражение пользователей и недовольство руководителей.

Поэтому в системе 1С:Предприятие есть несколько объектов конфигурации, которые позволяют создавать в базе данных структуры, предназначенные для накопления информации в удобном для последующего анализа виде.

Использование таких «хранилищ» данных позволяет нам, с одной стороны, накапливать в них данные, поставляемые различными документами (или другими объектами базы данных), а с другой стороны, легко создавать нужные нам отчеты или использовать эти данные в алгоритмах работы конфигурации (рис. 4.1).

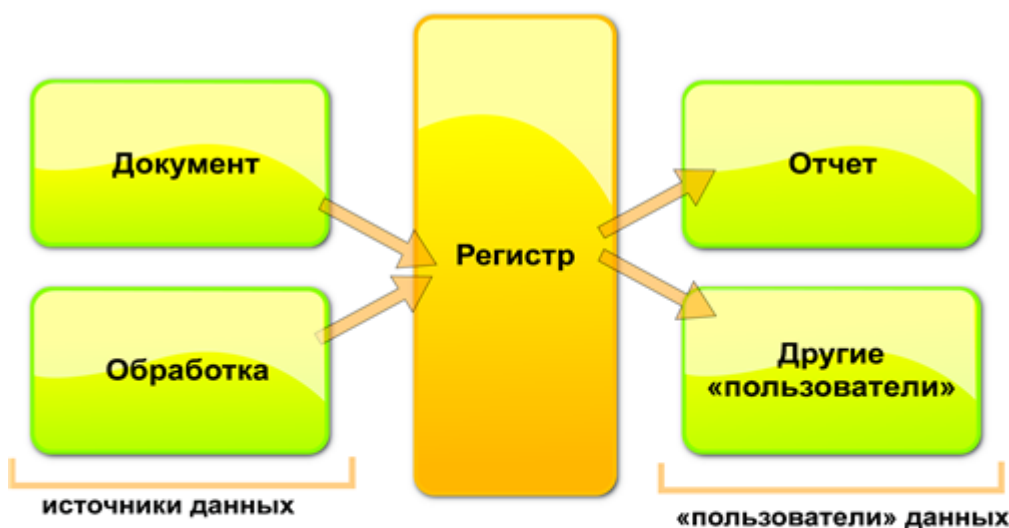


Рис. 4.1. Алгоритм работы конфигурации

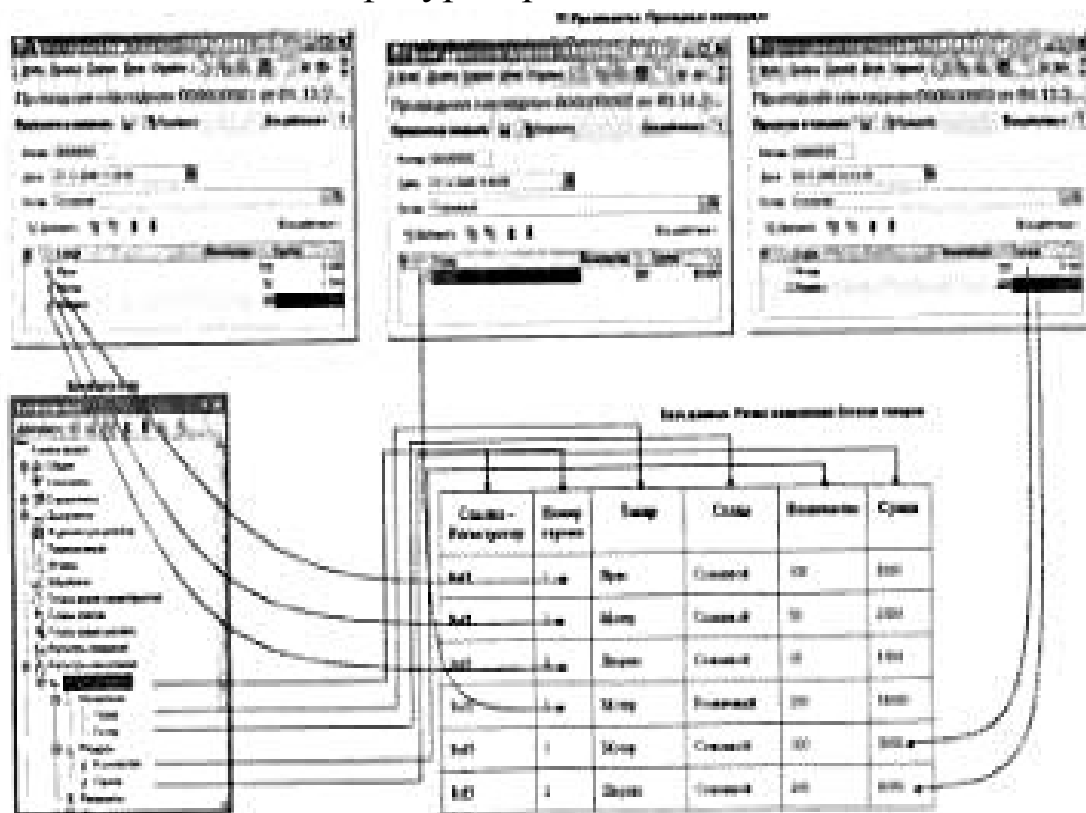
В конфигурации существует несколько объектов, называемых регистрами, для описания подобных «хранилищ». Сейчас мы рассмотрим один из них.

4.2. Что такое регистр накопления

Объект конфигурации Регистр накопления является прикладным объектом и предназначен для описания структуры накопления

данных. На основе объекта конфигурации Регистр накопления платформа создает в базе данных информационную структуру, в которой будут накапливаться данные, «поставляемые» различными объектами базы данных. Эти данные будут храниться в регистре в виде отдельных записей, каждая из которых имеет одинаковую, заданную в конфигураторе, структуру.

Виды числовой информации, накапливаемой регистром накопления, называются ресурсами и также являются подчиненными объектами и описываются в конфигураторе.



Изменение состояния регистра накопления происходит, как правило, при проведении документа, и заключается в том, что в регистр добавляется некоторое количество записей. Каждая запись содержит значения измерений, значения приращений ресурсов, ссылку на документ, который вызвал эти изменения (регистратор) и «направление» приращения (приход или расход). Такой набор записей называется движением регистра накопления. Каждому движению регистра накопления всегда должен соответствовать регистратор.

Кроме того, регистр накопления может хранить дополнительную информацию, описывающую каждое движение. Набор такой дополни-

тельной информации задается разработчиком при помощи реквизитов объекта конфигурации Регистр накопления.

4.3. Регистр накопления (Accumulation Register)

http://v8.1c.ru/overview/Term_000000176.htm

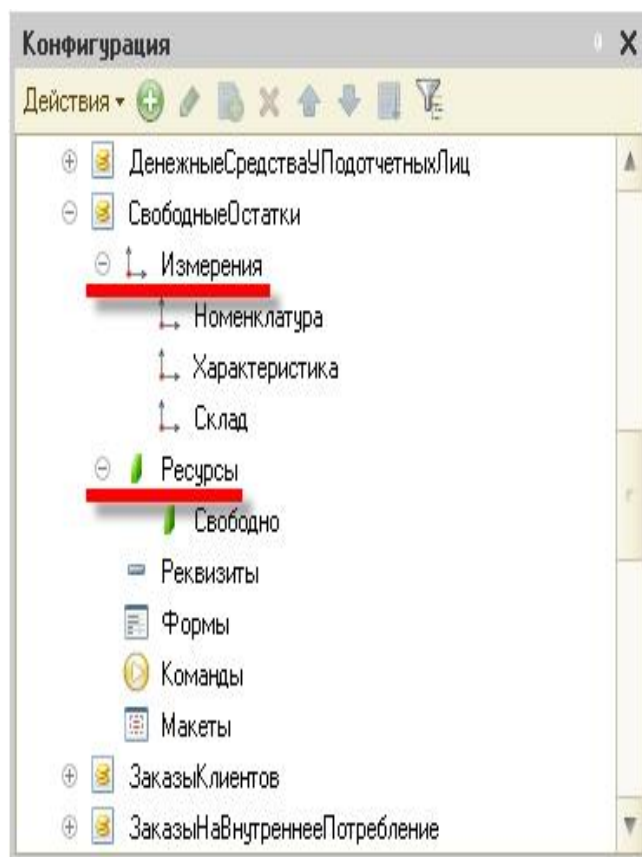
Регистры накопления - это прикладные объекты конфигурации. Они составляют основу механизма учета движения средств (финансов, товаров, материалов и т.д.), который позволяет автоматизировать такие направления, как складской учет, взаиморасчеты, планирование.

Регистр накопления образует многомерную систему измерений и позволяет "накапливать" числовые данные в разрезе нескольких измерений. Например, в таком регистре можно накапливать информацию об остатках товаров в разрезе номенклатуры и склада, или информацию об объемах продаж в разрезе номенклатуры и подразделения компании.

Структура

Информация в регистре накопления хранится в виде записей, каждая из которых содержит значения измерений и соответствующие им значения ресурсов.

Измерения регистра описывают разрезы, в которых хранится информация, а в **ресурсах** регистра накапливаются нужные числовые данные. Например, для регистра **Свободные остатки**, который имеет следующую структуру:



записи, производящие изменение ресурсов регистра в базе данных, будут выглядеть следующим образом:

Свободные остатки

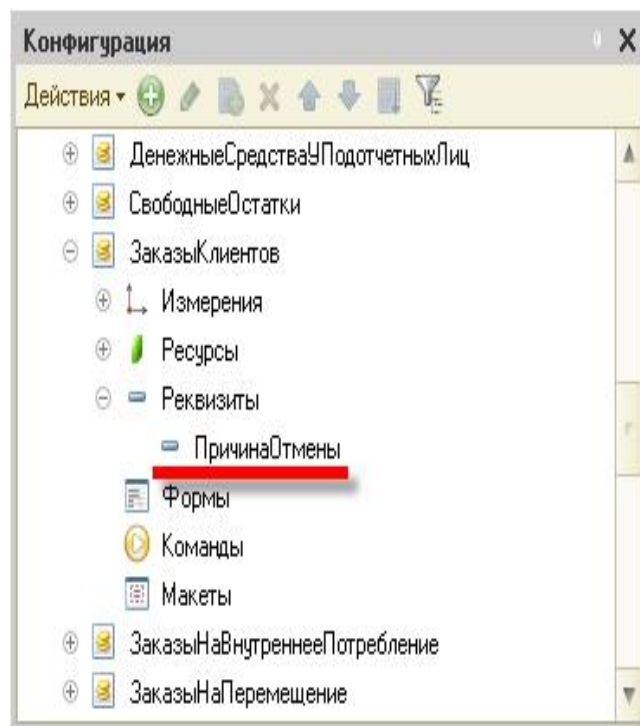
Найти... Все действия

Период	Регистратор	Но	Номенклатура	Характеристика	Склад	Свободно
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	4	Кресла	Светлый, дерево	Ясенево (склад)	5,000
+ 29.06.2009 14:06:55	Приходный ордер на товары 00000000011 от...	1	Набор мебели (под заказ)	Светлый, дерево	Ясенево (склад)	2,000
- 29.06.2009 14:49:44	Заказ на перемещение 00000000003 от 29.0...	1	Набор мебели (под заказ)	Светлый, дерево	Ясенево (склад)	2,000
+ 29.06.2009 15:03:06	Приходный ордер на товары 00000000012 от...	1	Набор мебели (под заказ)	Светлый, дерево	Центр (склад)	2,000
+ 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	1	Детский праздничный набор		Торговый зал	10,000
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	2	Грильяж (конфеты)		Торговый зал	2,000
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	3	Мишка (конфеты)		Торговый зал	2,000
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	4	Принц (печенье)		Торговый зал	10,000
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	5	Барбарис (конфеты)		Торговый зал	2,000
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	1	ВОСН	2-х камерный, "п...	Центр (склад)	1,000
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	2	СТИНОЛ 101	белый, 40*120*90...	Центр (склад)	1,000
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	3	Пылесос "Омега" 1250вт		Центр (склад)	1,000
- 06.07.2009 17:32:32	Возврат товаров поставщику 00000000001 о...	1	Ассорти (конфеты)		Торговый зал	10,000
- 06.07.2009 17:32:32	Возврат товаров поставщику 00000000001 о...	2	Принц (печенье)		Торговый зал	5,000

Поскольку регистр накопления служит для накопления числовых значений, каждая запись выполняет изменение хранимых ресурсов - движение. Движения, в общем случае, могут

либо добавлять некоторые приращения к хранимым ресурсам, либо отнимать их. Если должно выполняться увеличение хранимых ресурсов, - такое движение называется движением прихода ("+"), если уменьшение хранимых ресурсов - движением расхода ("-").

Вместе с каждой записью, находящейся в регистре накопления, можно хранить дополнительную произвольную информацию. Для этого служат реквизиты регистра накопления.



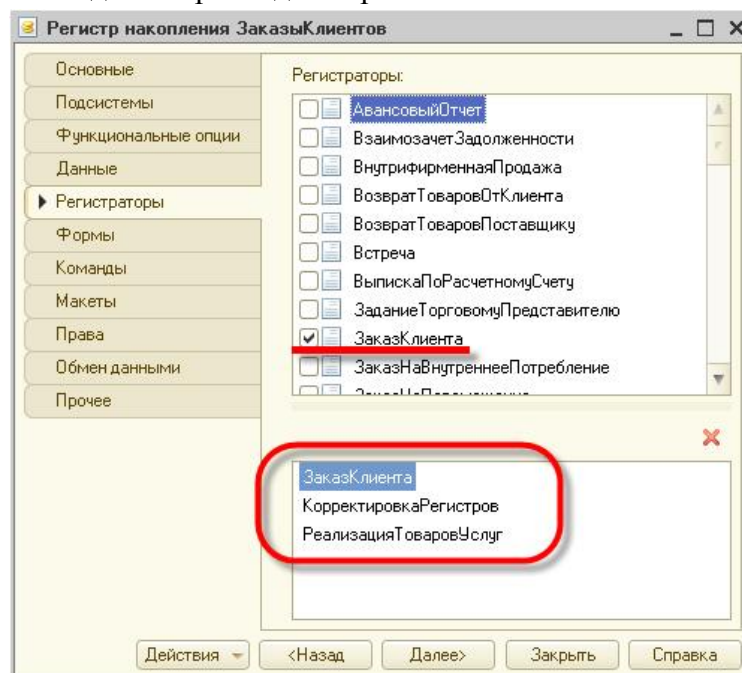
Связь с регистратором

Изменение состояния регистра накопления происходит, как правило, при проведении документа. Поэтому каждая запись регистра связана с определенным документом - регистратором, номером строки этого документа, и датой — периодом:

Период	Регистратор	Номер строки	Номенклатура
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	1	Стол обеденный
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	2	Стулья
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	3	Диван
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	4	Кресла
+ 29.06.2009 14:06:55	Приходный ордер на товары 00000000011 от...	1	Набор мебели (под за
- 29.06.2009 14:49:44	Заказ на перемещение 00000000003 от 29.0...	1	Набор мебели (под за
+ 29.06.2009 15:03:06	Приходный ордер на товары 00000000012 от...	1	Набор мебели (под за
+ 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	1	Детский праздничный
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	2	Трильяж (конфеты)
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	3	Мишка (конфеты)
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	4	Принц (печенье)
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	5	Барбарис (конфеты)
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	1	БОСХ
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	2	СТИНОЛ 101
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	3	Пылесос "Омега" 125
- 06.07.2009 17:32:32	Возврат товаров поставщику 00000000001 о...	1	Ассорти (конфеты)
- 06.07.2009 17:32:32	Возврат товаров поставщику 00000000001 о...	2	Принц (печенье)

В общем случае значение поле **Период** может не совпадать с датой документа. Например, документ **План продаж** может внести в регистр накопления записи о предполагаемых продажах компании несколькими разными датами.

Состав документов, которые могут создавать записи в регистре накопления, задается разработчиком в процессе создания прикладного решения:



Конструктор движений

Алгоритмы, по которым формируются записи в регистре, описываются средствами встроенного языка в процедурах соответствующих документов. Система содержит [конструктор движений](#), который помогает разработчику создавать алгоритмы проведения документа.

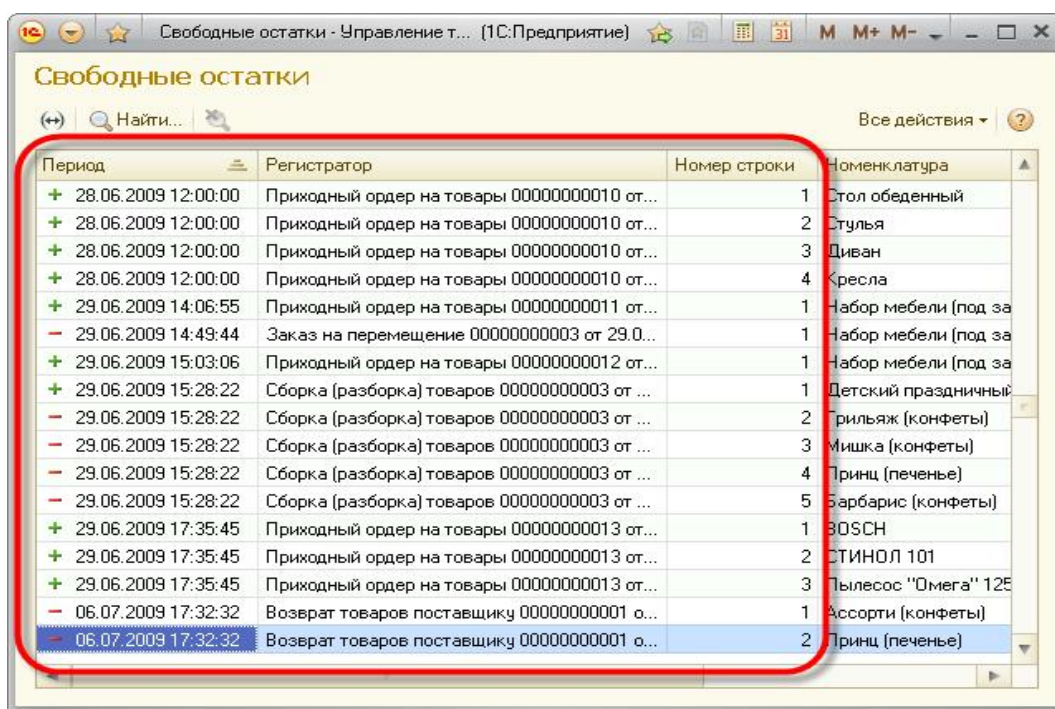
Уникальность записей

Система обеспечивает контроль уникальности записей, хранящихся в регистре накопления. Благодаря этому в регистре накоплений не может находиться двух записей, относящихся к одной и той же строке одного и того же документа.

Регистры остатков и регистры оборотов

Существует два вида регистров накопления: регистры накопления остатков и регистры накопления оборотов. Регистр накопления остатков позволяет хранить как итоговые значения ресурсов - остатки, так и изменения этих ресурсов - обороты. Регистр накопления оборотов является более "специализированным" видом регистра накопления и позволяет хранить только изменения ресурсов - обороты.

Существование регистра накопления оборотов связано с тем, что при автоматизации экономической деятельности существует большое количество ситуаций, когда требуется накапливать только обороты, а значения остатков не имеют смысла. Типичным примером использования регистра накопления оборотов является регистр **Выручка и себестоимость продаж**, хранящий информацию об объемах продаж:



Период	Регистратор	Номер строки	Номенклатура
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	1	Стол обеденный
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	2	Стулья
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	3	Диван
+ 28.06.2009 12:00:00	Приходный ордер на товары 00000000010 от...	4	Кресла
+ 29.06.2009 14:06:55	Приходный ордер на товары 00000000011 от...	1	Набор мебели (под за
- 29.06.2009 14:49:44	Заказ на перемещение 00000000003 от 29.0...	1	Набор мебели (под за
+ 29.06.2009 15:03:06	Приходный ордер на товары 00000000012 от...	1	Набор мебели (под за
+ 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	1	Детский праздничный
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	2	Трильяж (конфеты)
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	3	Мишка (конфеты)
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	4	Принц (печенье)
- 29.06.2009 15:28:22	Сборка (разборка) товаров 00000000003 от ...	5	Барбарис (конфеты)
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	1	БОSCH
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	2	СТИНОЛ 101
+ 29.06.2009 17:35:45	Приходный ордер на товары 00000000013 от...	3	Пылесос "Омега" 12E
- 06.07.2009 17:32:32	Возврат товаров поставщику 00000000001 о...	1	Ассорти (конфеты)
- 06.07.2009 17:32:32	Возврат товаров поставщику 00000000001 о...	2	Принц (печенье)

Поскольку регистр накопления оборотов не накапливает остатки ресурсов, для него не имеет смысла "направление" движения ресурсов (приход или расход); накапливается только величина изменения ресурсов. Поэтому все записи регистра накопления оборотов отмечены одинаковыми пиктограммами.

Агрегаты

Для оборотных регистров накопления платформа поддерживает специальный механизм агрегатов, который позволяет значительно ускорить получение данных из регистров, содержащих большое количество записей - сотни тысяч и миллионы записей. [Подробнее...](#)

Форма списка и форма набора записей

Для того чтобы пользователь мог просматривать данные, содержащиеся в регистре накопления, система поддерживает форму представления регистра накопления - форму списка. Она позволяет выполнять сортировку и отбор отображаемой информации по нескольким критериям:

Период	Регистратор	Номер стр...	Номенклатура	Характеристика	Склад
- 06.05.2009 12:00:00	Заказ клиента 00000000005 от 06.05.2009 12...	1	BOSCH	2-х камерный, "...	Центр
- 06.05.2009 12:00:00	Заказ клиента 00000000005 от 06.05.2009 12...	2	СТИНОЛ 101	белый, 40*120*9...	Центр
- 06.05.2009 12:00:00	Заказ клиента 00000000005 от 06.05.2009 12...	3	Пылесос "Омега...		Центр
- 07.05.2009 12:00:00	Заказ клиента 00000000006 от 07.05.2009 12...	1	Пылесос "Омега...		Центр
- 07.05.2009 12:00:00	Заказ клиента 00000000006 от 07.05.2009 12...	2	BOSCH	2-х камерный, к...	Центр
- 10.05.2009 12:00:00	Заказ клиента 00000000011 от 10.05.2009 12...	1	Ботинки женски...	35, светлый, ис...	Центр
- 10.05.2009 12:00:00	Заказ клиента 00000000011 от 10.05.2009 12...	2	Ботинки мужские	40, темный, нат...	Центр
- 10.05.2009 12:00:00	Заказ клиента 00000000011 от 10.05.2009 12...	3	Кондиционер EL...		Центр
- 10.05.2009 12:00:00	Заказ клиента 00000000011 от 10.05.2009 12...	4	Чайник BINATO...		Центр
+ 10.05.2009 12:00:00	Перемещение товаров 00000000001 от 10.05...	1	Кроссовки "ADI...	38, светлый, ис...	Торгов...
+ 10.05.2009 12:00:00	Перемещение товаров 00000000001 от 10.05...	2	Кроссовки "RE...	38, светлый, ис...	Торгов...
- 10.05.2009 15:31:00	Заказ на перемещение 00000000001 от 10.0...	1	Кроссовки "ADI...	38, светлый, ис...	Центр
- 10.05.2009 15:31:00	Заказ на перемещение 00000000001 от 10.0...	2	Кроссовки "RE...	38, светлый, ис...	Центр
- 11.05.2009 12:00:00	Заказ клиента 00000000012 от 11.05.2009 12...	1	Кондиционер FI...		Торгов...

Система может автоматически генерировать эту форму. Наряду с этим разработчик имеет возможность создать собственные формы, которые система будет использовать вместо формы умолчанию, в том числе и форму набора записей, которая позволяет добавлять, изменять и удалять записи регистра сведений:

N	Регистратор	Период	Вид движения	Номенклатура	Характеристика
1	Заказ клиента 00000000006 о...	07.05.2009 12:00:00	Расход	Пылесос "Омега" 1250вт	
2	Заказ клиента 00000000006 о...	07.05.2009 12:00:00	Расход	BOSCH	2-х камерный, красный, серт

Функциональные возможности регистра накопления

Основными функциональными возможностями, которые предоставляет регистр накопления разработчику, являются:

- выбор записей в заданном интервале по заданным критериям;
- выбор записей по регистратору;
- получение остатков и оборотов на указанный момент времени по заданным значениям измерений;
- режим работы с разделением итогов, который обеспечивает более высокую параллельность записи в регистр;
- отключение использования текущих итогов;
- расчет итогов на указанную дату;
- чтение, изменение и запись набора записей в регистр;
- возможность записи в регистр без пересчета итогов;
- полный пересчет итогов и пересчет итогов за указанный период.

4.4. Создание регистра накопления

Прежде всего, нас интересует информация о том, сколько и каких материалов есть у нас на складах. Для накопления такой информации мы создадим регистр «ОстаткиМатериалов».

150. Создайте регистр: выберите в дереве объектов конфигурации ветвь **Регистр накопления, МП**, выберите **Добавить**, в поле **Имя** введите **ОстаткиМатериалов**, нажмите **tab** и в поле **Синоним** должно появиться **Остатки материалов**, поле **Расширенное представление списка** введите **Движения по регистру Остатки материалов**, выберите **Далее**.

151. На вкладке подсистемы выберите **Учет материалов, Оказание услуг и Бухгалтерия**.

152. Выберите вкладку **Данные**, выделите ветвь **Измерения**, выберите **Добавить**, в поле **Имя** введите **Материал**, в списке **Тип** выберите **СправочникСсылка.Номенклатура**.

153. Создайте измерение: выберите **Измерения, МП**, выберите **Добавить**, в поле **Имя** введите **Склад**, в списке **Тип** выберите **СправочникСсылка.Склады**.

154. Создайте ресурс: выберите ветвь **Ресурсы, МП**, выберите **Добавить**, в поле **Имя** введите **Количество**, в списке **Тип** выберите **Число**, в поле **Длина** выберите **15**, **Точность** – **3**.

4.5. Создание движений документа

155. Создайте движение: выберите на дереве объектов конфигурации ветвь **Документы**, выберите **Приходная Накладная, М2**, выбо-

рите **Движения**, выберите **Регистры накопления**, выберите **ОстаткиМатериалов**, выберите **Конструктор движений**, выберите **Тип движения регистра** — **Приход**, в поле **Табличная часть** выберите табличную часть документа **Материалы**, нажмите **Заполнить выражение**, в качестве материала в регистр будет записан материал из табличной части документа, в качестве склада – склад, указанный в шапке документа, а в качестве количества – количество из табличной части документа, нажмите **ОК**.

Конструктор создал обработчик события «ОбработкаПроведения» объекта конфигурации Документ и поместил его в модуль объекта.

Внутри обработчика расположен цикл, который предназначен для перебора строк табличной части нашего документа. В цикле обращение к табличной части документа происходит по имени («Материалы»), а строки табличной части документа представляют собой коллекцию значений, для перебора которой можно использовать конструкцию **Для каждого ... из ... цикл**.

Объект встроенного языка ДокументОбъект имеет свойство «Движения». Оно возвращает коллекцию наборов записей регистров, которые принадлежат этому документу. К набору записей документа, принадлежащему конкретному регистру, можно обратиться, указав через точку имя этого регистра.

Таким образом, в первой строке тела цикла мы добавляем к набору записей, который создает наш документ в регистре, новую запись и сохраняем ее в переменной «Движение».

Затем мы присваиваем нужные значения всем полям этой записи и после перебора всех строк документа (после завершения циклов) «одним махом» записываем в регистр «ОстаткиМатериалов» весь набор записей движений документа.

156. Сделайте доступной в панели действий разделов команду для просмотра записей регистра накопления: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы, МП, Все подсистемы**, в окне Все подсистемы слева в списке Подсистемы выберите подсистему **УчетМатериалов**, в группе Панель навигации.Обычное для команды **Остатки материалов** включите **видимость** и перетащите ее в группу **Панель навигации**. См. Также

157. Сделайте доступной и перетащите в группу панель **Навигации**. См. Также для подсистем **ОказаниеУслуг** и **Бухгалтерия** команду **Остатки материалов**.

158. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

159. Откройте документ: выберите раздел **Бухгалтерия**, выберите команду **Приходные накладные**.

160. Перепроведите накладные 1 и 2.

161. Откройте регистр: выберите **Остатки материалов**.

Мы видим, что при проведении приходных накладных появляются соответствующие записи в регистре накопления **Остатки материалов**.

4.6. Команда перехода к движениям в форме документа

162. Выберите на дереве объектов конфигурации ветвь **Документы, Приходная Накладная, Формы, Форма Документа, М2**, выберите вкладку **Командный интерфейс**, в разделе **Панель навигации** раскройте группу **Перейти**, для команды **Остатки материалов** установите видимость.

163. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

164. Откройте документ: выберите раздел **Бухгалтерия**, выберите команду **Приходные накладные**, выберите накладную 2.

В форме документа появилась панель навигации, в которой можем переходить к списку записей регистра **Остатки Материалов**, связанному с документом, и обратно к содержимому документа.

4.7. Создание движений документа Оказание услуги

165. Создайте движение: выберите на дереве объектов конфигурации ветвь **Документы**, выберите **Оказание Услуги, М2**, выберите **Движения**, выберите **Регистры накопления**, выберите **Остатки Материалов**, выберите **Конструктор движений**, выберите **Тип движения регистра — Расход**, в поле **Табличная часть** выберите табличную часть нашего документа **Перечень Номенклатуры**, нажмите **Заполнить** выражение.

166. Но поле **Материал** автоматически не заполнилось, так как имя поля табличной части **Номенклатура** не совпадает с именем измерения

регистра **Материал**. Поэтому выделите поле регистра **Материал** и в окне **Реквизиты документа** выберите **ТекСтрокаПереченьНоменклатуры.Номенклатура, М2**, нажмите **ОК**.

167. Выберите на дереве объектов конфигурации ветвь **Документы, ОказаниеУслуги, Формы, ФормаДокумента, М2**, выберите вкладку **Командный интерфейс**, в разделе **Панель навигации** раскройте группу **Перейти**, для команды **Остатки материалов** установите видимость.

168. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне **Реорганизация информации** выберите **Принять**.

169. Откройте документ: выберите раздел **Оказание услуг**.

170. Перепроведите документ **Оказание услуги**.

171. Откройте регистр: выберите **Остатки материалов**.

5-й день. Простой отчет

5.1. Что такое отчет

Объект конфигурации Отчет является прикладным объектом и предназначен для описания средств и алгоритмов, при помощи которых пользователь сможет получать необходимые ему выходные данные. Алгоритм формирования выходных данных описывается при помощи визуальных средств или с использованием встроенного языка. В реальной жизни объектам конфигурации Отчет соответствуют всевозможные таблицы выходных данных, сводных данных диаграммы и пр.

5.2. Создание отчета

Теперь у нас все готово для того, чтобы можно было получать выходные данные. Поэтому приступим к созданию отчета, который будет показывать нам приход, расход и остатки материалов.

172. Создадим отчет: выберите в дереве объектов конфигурации ветвь **Отчеты, МП**, выберите **Добавить**, в поле **Имя** введите **Материалы**, нажмите **tab** и в поле **Синоним** должно появиться **Отчеты**, нажмите **Открыть схему компоновки данных**.

173. В окне конструктора макета выберите тип макета **Схема компоновки данных**, нажмите **Готово**.

174. Добавьте новый набор данных – запрос: нажмите кнопку **Добавить**, выберите **Добавить набор данных – запрос**.

175. Создайте текст запроса: нажмите кнопку **Конструктор запроса**.

Конструктор обладает большим количеством возможностей для визуального проектирования отчетов, но мы сейчас воспользуемся только самыми простыми его возможностями и просто определим те данные, которые хотим видеть в результате работы нашего отчета.

176. В списке «База данных» представлен состав объектов базы данных; на основе их данных мы имеем возможность построить отчет. Раскройте ветку **РегистрыНакопления** то мы увидим, что кроме таблицы регистра «ОстаткиМатериалов» в этой ветке присутствуют еще несколько виртуальных таблиц, которые формирует система.

177. Поскольку мы хотим видеть как остатки материалов, так и информацию об их поступлении и расходовании, поэтому раскройте виртуальную таблицу **ОстаткиМатериалов. ОстаткиИОбороты**.

Как вы видите, эта таблица содержит материал, склад и кроме этого начальные и конечные остатки, а также значения прихода, расхода и оборотов для всех ресурсов регистра «ОстаткиМатериалов».

178. Начнем выбирать поля таблицы в нужном нам порядке двойным щелчком мыши. Выберите **Склад, М2**, выберите **Материал, М2**, выберите **КоличествоНачальныйОстаток, М2**, **КоличествоПриход, М2**, выберите **КоличествоРасход, М2**, и в заключение **КоличествоКонечныйОстаток, М2**.

179. Нажмите **ОК**. Система автоматически сформирует формы и откроет их на экране.

180. Выберите вкладку **Настройки**, выделите корневой элемент **Отчет, МП**, выберите **Новая группировка**, нажмите **ОК**. Этим мы определили, что в отчет будут выводиться детальные записи из БД – записи, получаемые в результате выполнения запроса без итогов.

181. Настройте поля, которые будут выводиться в результат отчета: перейдите в нижнем окне настроек на вкладку **Выбранные поля** и перенесите мышью из списка доступных полей: **Склад, Материал, КоличествоНачальныйОстаток, КоличествоПриход, КоличествоРасход, КоличествоКонечныйОстаток**.

182. Выберите вкладку **Параметры**, выберите флажками параметры **Дата начала** и **Дата окончания**, затем выделим каждый их этих параметров, **МП**, выберите **Свойства элемента пользовательских настроек**, поставьте флажок **Включать в пользовательские настройки**.

183. Закройте конструктор схемы компоновки данных.

183. В окне редактирования объекта конфигурации **Отчет Материалы** выберите вкладку **Подсистемы**, выберите **Учет материалов, Оказание услуг и Бухгалтерия**.

184. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

185. Выберите **Учет материалов**, выберите команду **Материалы**, нажмите **Сформировать**.

6-й день. Макеты

6.1. Что такое макет

Объект конфигурации Макет предназначен для хранения различных форм представления данных, которые могут потребоваться каким либо объектам конфигурации или всему прикладному решению в целом. Макет может содержать табличный или текстовый документ, двоичные данные, HTML-документ или Active Document. Макеты могут существовать как сами по себе (общие макеты), так и быть подчинены какому либо объекту конфигурации.

Одно из предназначений макета, подчиненного объекту конфигурации и содержащего табличный документ - создание печатной формы этого объекта.

Создание печатной формы заключается в конструировании ее составных частей - именованных областей, из которых затем «собирается» готовая печатная форма. Порядок заполнения областей данными и порядок вывода их в итоговую форму описывается при помощи встроенного языка.

Печатная форма может включать в себя различные графические объекты: картинки, OLE-объекты, диаграммы и т.д.

Помимо создания макета «вручную», конфигуратор предоставляет разработчику возможность воспользоваться специальным инструментом - конструктором печати, который берет на себя большинство рутинной работы по созданию макета.

6.2. Создание макета документа

186. Выберите на дереве объектов объектов ветвь **Документы**, выберите **ОказаниеУслуги, M2**, выберите вкладку **Макеты**, выберите **Конструкторы**, **Конструктор печати**.

187. На первом шаге укажите, что будет создана новая команда Печать для формирования печатной формы документа выберите **Создать новую команду**, **Далее**.

188. На втором шаге нажатием **двойной** стрелки определим, что все реквизиты нашего документа будут отображены в шапке печатной формы, **Далее**.

189. На третьем шаге точно также определим, что **все** реквизиты табличной части документа будут отображены в печатной форме, Далее.

190. На четвертом шаге конструктор предложит сформировать нам подвал (нижнюю часть) печатной формы. Мы не станем ничего указывать (подвал в данном случае использовать не будем), и перейдем к пятому шагу, Далее.

191. Здесь ничего изменять не будем. Тем самым согласимся с тем, что команда для вызова процедуры формирования печатной формы будет помещена в командную панель формы, в раздел Важное. Нажмите **ОК**.

192. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

193. Откройте документ: выберите документ **Оказание услуги, М2**, нажмите **Печать**.

В данной форме не хватает итоговой суммы.

6.3. Редактирование макета документа

194. Выберите в дереве объектов конфигурации ветвь **Документы**, выберите **ОказаниеУслуги**, выберите вкладку **Макеты**, выберите **Печать, М2**, как видите макет документа состоит из именованных областей, которые в определенном порядке выводятся на печать.

195. Добавим новую область для вывода итоговой суммы документа: выделим мышью **две пустые строки** (например, 15 и 16) и выберите **Таблица, Имена, Имена**, введите **Всего**, нажмите **Присвоить, Заккрыть**.

196. Чтобы формат добавленных нами строк совпадал с имеющимся форматом заголовка и табличной части документа, изменим ширину колонок. Для этого потянем мышью в заголовке таблицы за правую границу колонки 2 так, чтобы ее ширина совпала с шириной колонки № в шапке документа. Отпустим мышь. Система предложит создать новый формат для выделенных строк. Согласитесь.

197. Аналогичные действия выполним для колонок 3, 4, 5, 6.

198. В созданной области в колонке **Цена** введите **Всего**, а в колонке **Сумма** введите **ВсегоПоДокументу**.

199. Вызвав палитру свойств для последней заполненной нами ячейки, укажем, что в этой ячейке будет находиться не текст, а параметр: выберите клетку с **ВсегоПоДокументу**, **МП**, **Свойства**, в списке **Заполнение** выберите **Параметр**.

Здесь следует сказать о том, что каждая ячейка редактируемого нами табличного документа может содержать либо текст, либо некоторый параметр, либо шаблон. Текст, содержащийся в ячейке, будет показан на экране. Параметр будет заменен некоторым значением, которое может быть присвоено ему средствами встроенного языка. Текст, содержащийся в ячейке, является именем этого параметра. Шаблон представляет собой текстовую строку, в определенные места которой будут вставлены значения параметров.

Поэтому, указав для ячейки в качестве заполнения «Параметр», мы определили параметр области с именем «ВсегоПоДокументу», которому присвоим нужное нам значение при формировании печатной формы.

200. Откроем модуль менеджера документа **ОказаниеУслуги**: выберите вкладку **Прочее**, нажмите **Модуль менеджера** и после цикла добавьте в нее следующий код:

```
ОбластьЗаголовков = Макет.ПолучитьОбласть("Заголовков");
```

```
Шапка = Макет.ПолучитьОбласть("Шапка");
```

```
ОбластьПереченьНоменклатурыШапка =
```

```
Макет.ПолучитьОбласть("ПереченьНоменклатурыШапка");
```

```
ОбластьПереченьНоменклатуры =
```

```
Макет.ПолучитьОбласть("ПереченьНоменклатуры");
```

```
ОбластьИтог = Макет.ПолучитьОбласть("Всего");
```

```
ТабДок.Очистить();
```

```
ВставлятьРазделительСтраниц = Ложь;
```

```
Пока Выборка.Следующий() Цикл
```

```
    Если ВставлятьРазделительСтраниц Тогда
```

```
        ТабДок.ВывестиГоризонтальныйРазделительСтраниц();
```

```
    КонецЕсли;
```

```
ТабДок.Вывести(ОбластьЗаголовков);
```

```
Шапка.Параметры.Заполнить(Выборка);  
ТабДок.Вывести(Шапка, Выборка.Уровень());
```

```
ТабДок.Вывести(ОбластьПереченьНоменклатурыШапка);  
ВыборкаПереченьНоменклатуры =  
Выборка.ПереченьНоменклатуры.Выбрать();  
СуммаИтог=0;  
Пока ВыборкаПереченьНоменклатуры.Следующий() Цикл
```

```
ОбластьПереченьНоменклатуры.Параметры.Заполнить(ВыборкаПере  
ченьНоменклатуры);  
ТабДок.Вывести(ОбластьПереченьНоменклатуры,  
ВыборкаПереченьНоменклатуры.Уровень());
```

```
СуммаИтог=СуммаИтог+ВыборкаПереченьНоменклатуры.Сумм  
а;
```

```
КонецЦикла;
```

```
ОбластьИтог.Параметры.ВсегоПоДокументу =  
СуммаИтог;
```

```
ТабДок.Вывести(ОбластьИтог);
```

```
ВставлятьРазделительСтраниц = Истина;
```

```
КонецЦикла;
```

Смысл добавления следующий. Обращаемся к макету документа ОказаниеУслуги по его имени – Макет.

Используя его метод ПолучитьОбласть() получаем область Всего и сохраняем ее в переменной ОбластьИтог.

В цикле обхода строк табличной части документа, полученных в результате выполнения запроса, накапливаем в переменной СуммаИтог значение суммы табличной части документа по колонке Сумма.

Затем обращаемся к параметру ВсегоПоДокументу находящемуся в области Всего и присваиваем ему значение переменной СуммаИтог.

В итоге выводим итоговую область в табличный документ, который будет показан на экране и распечатан пользователем ТабДок.Вывести(ОбластьИтог).

201. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

202. Откройте документ: выберите документ **Оказание услуги, М2**, нажмите **Печать**, должна быть строка **Всего** и цифровая сумма.

6.4. Редактирование формы

А теперь, для того, чтобы наш документ **ОказаниеУслуги**, выглядел вполне законченным, добавим итоговую сумму по документу и на экранную форму, чтобы пользователь мог видеть ее в процессе заполнения табличной части документа.

203. Выберите на дереве объектов конфигурации ветвь **Документы**, выберите **ОказаниеУслуги, Формы, ФормаДокумента, М2**, выберите **ПереченьНоменклатуры, М2**.

204. Откроем палитру свойств для табличного поля, расположенного в форме, и установим флажком свойство **«Подвал»**.

205. Выберите **ПереченьНоменклатурыЦена, М2**, в поле текст подвала области **Основные** введите **Всего:**, в списке **Горизонтальное положение в подвале** выберите **Право**, в свойстве **Шрифт подвала** изменим начертание на **Жирный**.

206. Выберите **ПереченьНоменклатурыСумма, М2**, в списке **Горизонтальное положение в подвале** выберите **Право**, в свойстве **Шрифт подвала** тоже изменим начертание на **Жирный**.

207. Для того чтобы в подвале колонки **Сумма** отображался итог по ней, выберите **три точки** в поле **ПутьКДаннымПодвала**, выберите **Объект, ПереченьНоменклатуры, ИтогСумма**, нажмите **ОК**.

208. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

209. Откройте документ: выберите документ **Оказание услуги, М2**, должна быть строка **Всего** и цифровая сумма.

7-й день. Периодические регистры сведений

7.1. Регистр сведений (Information Register)

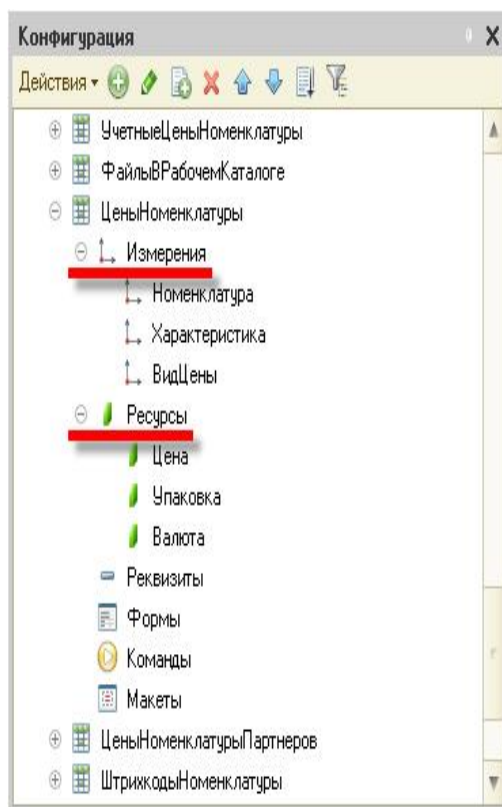
Регистры сведений - это прикладные объекты конфигурации. Они позволяют хранить в прикладном решении произвольные данные в разрезе нескольких измерений. Например, в регистре сведений можно хранить курсы валют в разрезе валют, или цены предприятия в разрезе номенклатуры и типа цен.

Структура

Информация в регистре сведений хранится в виде записей, каждая из которых содержит значения измерений и соответствующие им значения ресурсов.

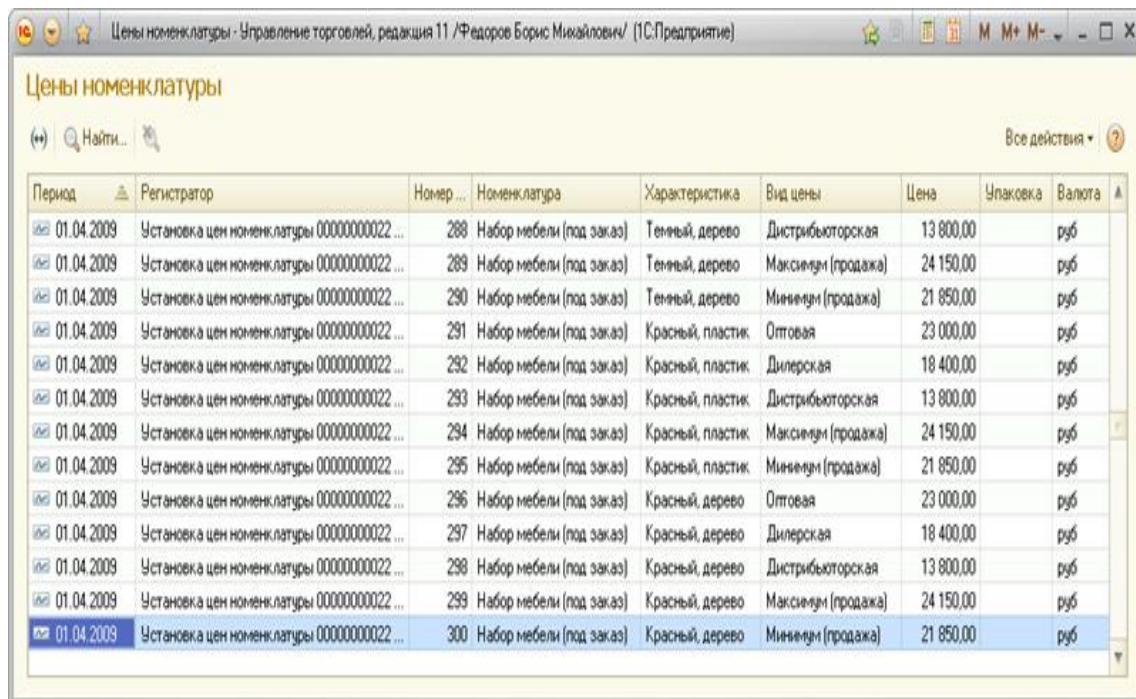
Измерения регистра описывают разрезы, в которых хранится информация, а **ресурсы** регистра непосредственно содержат хранимую информацию. Например, для регистра сведений **Цены номенклатуры**, который имеет следующую структуру:

записи, хранимые в базе данных, будут выглядеть следующим образом:



Таким образом, мы можем хранить информацию о том, что, скажем, дистрибьюторская цена на набор мебели из темного дерева равна 13800 рублей, а оптовая цена на такой же набор, но из красного дерева, равна 23000 рублей.

Вместе с каждой записью, находящейся в регистре сведений, можно хранить дополнитель-

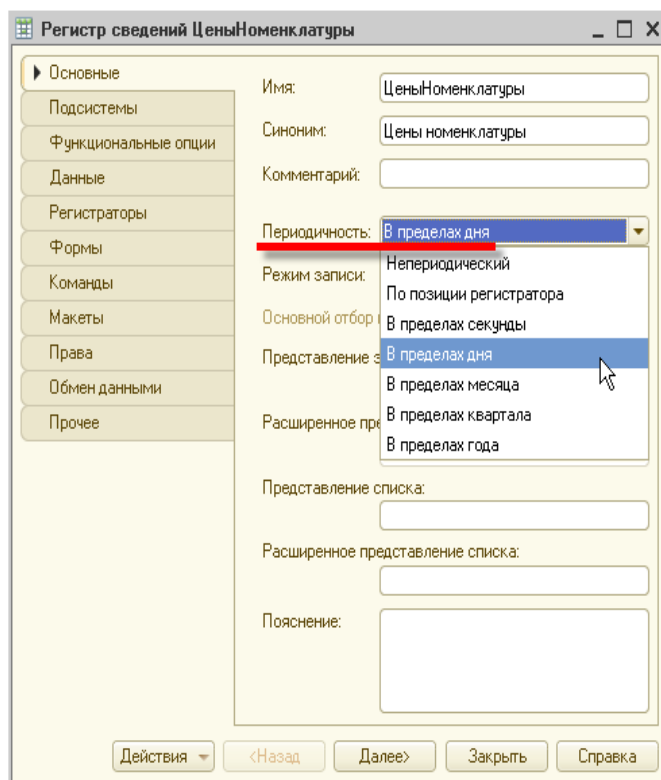


Период	Регистратор	Номер ...	Номенклатура	Характеристика	Вид цены	Цена	Упаковка	Валюта
01.04.2009	Установка цен номенклатуры 00000000022 ...	288	Набор мебели (под заказ)	Темный, дерево	Дистрибьюторская	13 800,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	289	Набор мебели (под заказ)	Темный, дерево	Максимум (продажа)	24 150,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	290	Набор мебели (под заказ)	Темный, дерево	Минимум (продажа)	21 850,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	291	Набор мебели (под заказ)	Красный, пластик	Оптовая	23 000,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	292	Набор мебели (под заказ)	Красный, пластик	Дилерская	18 400,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	293	Набор мебели (под заказ)	Красный, пластик	Дистрибьюторская	13 800,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	294	Набор мебели (под заказ)	Красный, пластик	Максимум (продажа)	24 150,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	295	Набор мебели (под заказ)	Красный, пластик	Минимум (продажа)	21 850,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	296	Набор мебели (под заказ)	Красный, дерево	Оптовая	23 000,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	297	Набор мебели (под заказ)	Красный, дерево	Дилерская	18 400,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	298	Набор мебели (под заказ)	Красный, дерево	Дистрибьюторская	13 800,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	299	Набор мебели (под заказ)	Красный, дерево	Максимум (продажа)	24 150,00		руб
01.04.2009	Установка цен номенклатуры 00000000022 ...	300	Набор мебели (под заказ)	Красный, дерево	Минимум (продажа)	21 850,00		руб

ную произвольную информацию. Для этого служат реквизиты регистра сведений.

Периодичность

Одной из возможностей регистра сведений является хранение данных не только в разрезе указанных измерений, но и в разрезе времени. Разработчик может указать минимальную периодичность, с которой записи будут заноситься в регистр:



Регистр сведений ЦеныНоменклатуры

Имя: ЦеныНоменклатуры

Синоним: Цены номенклатуры

Комментарий:

Периодичность: В пределах дня

Режим записи: Непериодический

Основной отбор: По позиции регистратора

Представление: В пределах дня

Расширенное пр: В пределах месяца, В пределах квартала, В пределах года

Представление списка:

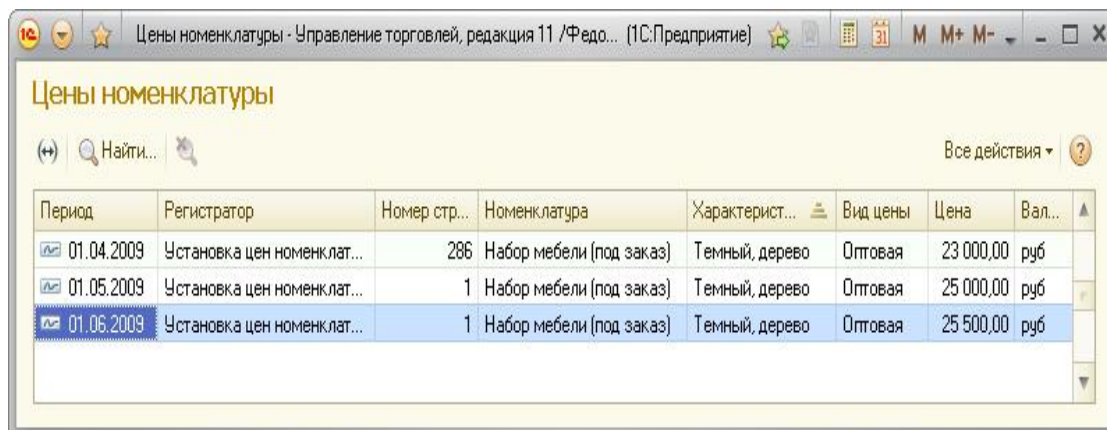
Расширенное представление списка:

Пояснение:

Действия <Назад Далее> Закреть Справка

В этом случае к каждой записи регистра будет добавляться поле **Период**, хранящее дату, которой были внесены записи в регистр. Использование периодичности регистра сведений позволяет не просто хранить статические данные, но и отслеживать их изменение во времени.

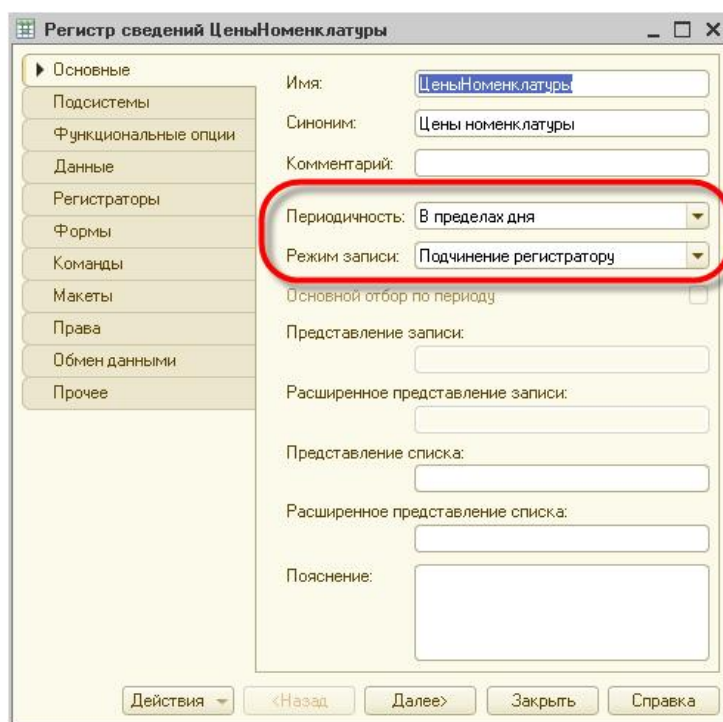
Например, периодический регистр сведений **Цены номенклатуры** может не только хранить информацию о том, какова цена на определенную номенклатуру сейчас, но и о том, как она изменялась в прошлом (или будет изменяться в будущем):



Период	Регистратор	Номер стр...	Номенклатура	Характерист...	Вид цены	Цена	Вал...
01.04.2009	Установка цен номенклат...	286	Набор мебели (под заказ)	Темный, дерево	Оптовая	23 000,00	руб
01.05.2009	Установка цен номенклат...	1	Набор мебели (под заказ)	Темный, дерево	Оптовая	25 000,00	руб
01.06.2009	Установка цен номенклат...	1	Набор мебели (под заказ)	Темный, дерево	Оптовая	25 500,00	руб

Подчинение регистратору

Внесение изменений в регистр сведений может выполняться как вручную, так и при помощи документов. В случае, когда изменения в регистр сведений вносятся с помощью документов, к каждой записи регистра добавляется специальное поле, в котором хранится информация о регистраторе - документе, с которым связана эта запись. В процессе создания прикладного решения разработчик указывает, какой именно режим записи будет использоваться данным регистром сведений:



Регистр сведений ЦеныНоменклатуры

Имя: ЦеныНоменклатуры

Синоним: Цены номенклатуры

Комментарий:

Периодичность: В пределах дня

Режим записи: Подчинение регистратору

Основной отбор по периоду

Представление записи:

Расширенное представление записи:

Представление списка:

Расширенное представление списка:

Пояснение:

Действия <Назад Далее> Закреть Справка

Использование режима записи **Подчинение регистратору** может потребоваться в случае, когда логика работы прикладного решения требует того, чтобы изменения, выполняемые в регистре сведений, были жестко связаны с документами, фиксирующими факты хозяйственной деятельности.

Например, изменение цен компании может производиться только определенным кругом лиц, и каждое такое изменение должно сопровождаться "бумажным" документом. В этом случае можно использовать режим подчинения регистратору, при котором изменение цен может быть выполнено только специальным документом - **Изменение цен номенклатуры**.

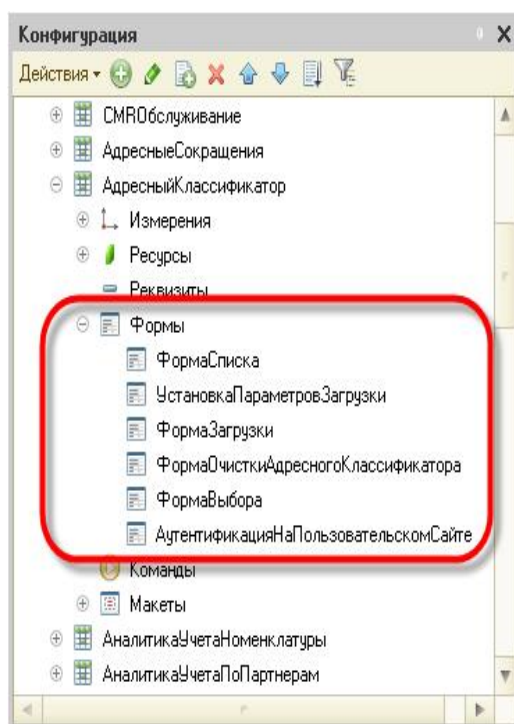
Уникальность записей

Система обеспечивает контроль уникальности записей, хранящихся в регистре сведений. Таким образом, в регистре сведений не может находиться двух одинаковых записей. Одинаковыми считаются записи, у которых совпадает ключ записи. Ключ записи формируется системой автоматически, на основании значений, содержащихся в полях записи, и зависит от вида регистра сведений.

В общем случае в формировании ключа записи будут участвовать значения регистратора, периода и значения измерений. Таким образом, например, в неперiodическом регистре сведений **Цены номенклатуры** с независимым режимом записи не может существовать двух записей о розничной цене конфет ассорти. Точно так же, как в периодическом регистре сведений **Цены номенклатуры**, подчиненном регистратору, не может существовать двух записей о розничной цене конфет ассорти, внесенных одной и той же датой, одним и тем же документом **Изменение цен номенклатуры**.

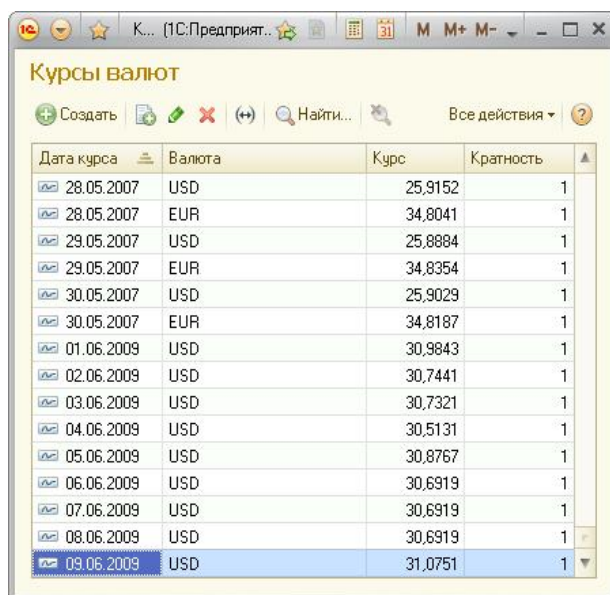
Формы

Для того чтобы пользователь мог просматривать и изменять данные, содержащиеся в регистре сведений, система поддерживает несколько форм представления регистра. Система может автоматически генерировать все нужные формы регистра. Наряду с этим разработчик имеет возможность создать собственные формы, которые система будет использовать вместо форм по умолчанию:



Форма списка

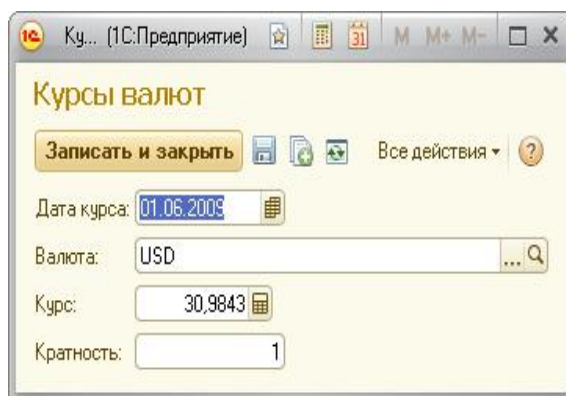
Для просмотра данных, содержащихся в регистре сведений, используется форма списка. Она позволяет выполнять навигацию по регистру, добавлять, помечать на удаление и удалять записи регистра. Форма списка позволяет выполнять сортировку и отбор отображаемой информации по нескольким критериям:



Дата курса	Валюта	Курс	Кратность
28.05.2007	USD	25,9152	1
28.05.2007	EUR	34,8041	1
29.05.2007	USD	25,8884	1
29.05.2007	EUR	34,8354	1
30.05.2007	USD	25,9029	1
30.05.2007	EUR	34,8187	1
01.06.2009	USD	30,9843	1
02.06.2009	USD	30,7441	1
03.06.2009	USD	30,7321	1
04.06.2009	USD	30,5131	1
05.06.2009	USD	30,8767	1
06.06.2009	USD	30,6319	1
07.06.2009	USD	30,6319	1
08.06.2009	USD	30,6319	1
09.06.2009	USD	31,0751	1

Форма записи

Для просмотра и изменения отдельных записей регистра сведений используется форма записи. Как правило, она представляет данные в удобном для восприятия и редактирования виде:



Курсы валют

Записать и закрыть Все действия

Дата курса: 01.06.2009

Валюта: USD

Курс: 30,9843

Кратность: 1

Функциональные возможности регистра сведений

Основными функциональными возможностями, которые предоставляет регистр сведений разработчику, являются:

- создание, изменение и удаление записей;
- выбор записей в заданном интервале по заданным критериям;
- выбор записей по регистратору;
- получение значений ресурсов записей, соответствующих указанному периоду и значениям измерений;
- получение значений ресурсов наиболее ранних и наиболее поздних записей регистра, соответствующих указанному периоду и значениям измерений.

7.2. Создание периодического регистра сведений

Создадим периодический регистр сведений, который будет хранить развернутые во времени розничные цены материалов и стоимости услуг, оказываемых нашим ООО.

210. Создадим новый объект конфигурации Регистр сведений: выберите на дереве объектов конфигурации ветвь **Регистр сведений, МП**, выберите **Добавить**, на вкладке **Основные** в поле **Имя** введите **Цены**, в списке периодичность выберите **в пределах секунды**, в поле **Представление записи** введите **Цена**, в поле **Представление списка** введите **Цены на номенклатуру**, выберите **Режим записи – независимый**, Далее.

211. На вкладке **Подсистемы** выберите **Учет материалов, Оказание услуг и Бухгалтерия**.

212. Выберите закладку **Данные**, выберите **Измерения**, нажмите **Добавить** и введите имя измерения регистра **Номенклатура** с типом **СправочникСсылка.Номенклатура**. Укажите, что это измерение будет **ведущим**.

Свойство "Ведущее" имеет смысл использовать лишь тогда, когда измерение имеет тип ссылки на объект базы данных. Установка свойства "Ведущее" будет говорить о том, что запись регистра сведений представляет интерес, только пока существует этот объект. При удалении объекта, все записи регистра сведений по этому объекту тоже будут автоматически удалены. Кроме того, в форме списка справочника появляется кнопка командной панели "Перейти", по которой возможен переход к записям регистра, отобранным по значению выбранного элемента справочника.

213. Создайте новый ресурс **Цена**, тип – **Число**, длина – **15**, точность – **2**, неотрицательное.

214. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации**

базы данных. Обновить конфигурацию базы данных? выберите **Да**, в окне Реорганизация информации выберите **Принять**.

215. Откройте регистр: выберите **Учет материалов**, выберите **Цены на номенклатуру**, нажмите **Создать**.

216. Введите стоимость услуг от **26 июля**: **Ремонт отечественного телевизора 600 руб, Ремонт импортного телевизора 800 руб, Подключение воды 800 руб, Подключение электричества 800 руб.**

217. Введите розничные цены на материалы от 26 июля: **Строчный трансформатор Samsung – 900, Строчный трансформатор GoldStar – 400, Транзистор Philips 2N2369 – 5, Шланг резиновый 150, Кабель электрический – 30.**

Поскольку цены хранятся с привязкой к дате, мы можем заранее установить новые цены и быть уверенными в том, что новые цены вступят в действие не раньше указанной для них даты.

7.3. Автоматическая подстановка цены в документе

218. Создайте функцию, которая будет возвращать актуальную розничную цену номенклатуры: в ветке **Общие, Общие модули** создайте новый объект конфигурации **Модуль** и назовите его **Работа-СоСправочниками**, по умолчанию установлен флажок **Сервер**. Это означает, что экземпляры этого модуля будут скомпилированы только на стороне сервера.

219. Установите флажок **Вызов сервера**, чтобы экспортные процедуры и функции этого модуля можно было вызывать с клиента.

220. Разместите код:

```
Функция РозничнаяЦена(АктуальнаяДата,  
ЭлементНоменклатуры) Экспорт  
    //создать вспомогательный объект Отбор  
    Отбор = Новый  
Структура("Номенклатура",ЭлементНоменклатуры);  
    //получить актуальные значения ресурсов регистра  
    ЗначенияРесурсов=РегистрыСведений.Цены.ПолучитьПослед  
нее(АктуальнаяДата, Отбор);  
    Возврат ЗначенияРесурсов.Цена;  
КонецФункции
```

Для получения розничной цены мы будем передавать в функцию два параметра:

АктуальнаяДата - параметр типа Дата, который будет определять точку на оси времени, на которую нас интересует значение розничной цены

ЭлементНоменклатуры - ссылка на элемент справочника "Номенклатура", для которого мы хотим получить розничную цену.

В теле процедуры мы создаем сначала вспомогательный объект Отбор, с помощью которого определяем, что нас будут интересовать записи регистра, в которых измерение "Номенклатура" равно переданной в процедуру ссылке на элемент справочника.

Во второй строке мы обращаемся к менеджеру регистра сведений "Цены" (РегистрыСведений.Цены) и выполняем метод ПолучитьПоследнее(), который возвращает нам значения ресурсов наиболее поздней записи регистра, которая соответствует передаваемой дате ("АктуальнаяДата") и значениям измерений регистра ("Отбор").

Значения ресурсов возвращаются в структуре, поэтому в следующей строке мы получаем искомую нами розничную цену просто указав имя нужного нам ресурса регистра через точку (ЗначенияРесурсов.Цена).

7.4. Автоматическое заполнение цены в документе ОказаниеУслуги

При создании документа ОказаниеУслуги нам необходимо обеспечить автоматическое заполнение поля Цена после того, как пользователь выберет услугу. Причем цена услуги должна определяться исходя из даты создаваемого документа.

221. Выберите на дереве объект **Документы**, выберите **ОказаниеУслуги**, выберите **Формы**, **ФормаДокумента**, **М2**, выберите **ПереченьНоменклатурыНоменклатура**, **М2**, найдем событие **При изменении**.

Нажмем на кнопку с лупой и в открывшейся заготовке обработчика события напишем следующий текст:

```
ПроцедураПереченьНоменклатурыНоменклатураПриИзменении(Элемент)
```

```
//получить текущую строку табличной части
```

```
СтрокаТабличнойЧасти =
```

```
Элементы.ПереченьНоменклатуры.ТекущиеДанные;
```

```
//установить цену
СтрокаТабличнойЧасти.Цена =
РаботаСоСправочниками.РозничнаяЦена(Объект.Дата,
СтрокаТабличнойЧасти.Номенклатура);
//пересчитать сумму строки
РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);
КонецПроцедуры
```

Первая строка обработчика вам уже знакома - мы получаем текущую строку табличной части документа, так как она нам понадобится в дальнейшем.

Во второй мы устанавливаем полученную цену в документе, вызывая нашу процедуру «РозничнаяЦена». Первым параметром мы передаем дату документа, на которую необходимо получить цену, а вторым параметром мы передаем ссылку, которую отображает элемент управления формой, вызвавший это событие (Элемент.Значение), т.е. ссылку на элемент справочника «Номенклатура».

В заключение мы вызываем нашу процедуру «РассчитатьСумму» из общего модуля «РаботаСДокументами» для того, чтобы она пересчитала итоговую сумму в строке нашего документа.

222. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

223. Откройте регистр: выберите Главное меню, Все функции. **Регистр сведений**, выберите **Цены**, нажмите **Создать**

224. Для транзистора **Philips** добавьте новую цену **7 руб.** от **30 июля**, нажмите **Записать и закрыть**.

227. Теперь откроем документ **Оказание услуги** Этим документом мы как раз «израсходовали» один такой транзистор. Установите дату документа **28 июля**, когда было задано первое значение цены транзистора, выберите **транзистор Philips** в колонке «Номенклатура» табличной части документа, цена должна автоматически стать **5 руб**

228. Теперь измените дату документа на **1 августа** и снова повторим выбор транзистора. Будет установлено новое значение цены: **7 руб.**

8-й день. Перечисления

8.1. Добавление перечисления

Перечисления - это прикладные объекты конфигурации. Они позволяют хранить в информационной базе наборы значений, которые не изменяются в процессе работы прикладного решения.

Мы создадим у справочника Номенклатура реквизит, тип значения которого образуется объектом конфигурации Перечисление. Это поможет нам определять, чем является элемент справочника Номенклатура: услугой или материалом.

229. Выберите на дереве объектов конфигурации ветвь **Перечисления, МП, Добавить**, в поле **Имя** введите **ВидыНоменклатуры**.

230. Выберите вкладку **Данные**, нажмите **Добавить**, введите значения **Материал** и **Услуга**.

231. Затем добавим в справочник новый реквизит: выберите **Справочники, Номенклатура**, введите реквизит **ВидНоменклатуры** с типом **ПеречислениеСсылка.ВидыНоменклатуры**.

232. Выберите **Отладка, Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне **Реорганизация информации** выберите **Принять**.

233. При сообщении о том, что наше перечисление не включено нив одну подсистему игнорируем его.

234. Откройте справочник **Номенклатура**, задайте для каждого элемента справочника Номенклатура соответствующее значение реквизита ВидНоменклатуры.

8.2. Изменение процедуры проведения документа

Когда создавались движения документа «ОказаниеУслуги» по регистру накопления «ОстаткиМатериалов», мы сказали, что они не совсем правильные, поскольку в регистр будут попадать не только записи об израсходованных материалах, но и записи об оказанных услугах.

Теперь мы займемся тем, что доработаем документ таким образом, чтобы в регистре появлялись только записи, относящиеся к

расходу материалов. Эта доработка будет не совсем эффективна с точки зрения производительности, зато позволит нам получить нужные данные в регистре «ОстаткиМатериалов».

235. Скорректируем движения документа, исключив из обработки те строки табличной части, в которых находятся услуги. Для этого в обработчик события «ОбработкаПроведения», расположенный в модуле документа «ОказаниеУслуги», добавим следующий текст (добавленный текст выделен жирным шрифтом): в окне **Конфигуратора** выберите на дереве объект **Документы**, выберите **ОказаниеУслуги, МП**, выберите **Открыть Модуль объекта**, добавьте строки кода

```
Процедура ОбработкаПроведения(Отказ, Режим)
  //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
  // Данный фрагмент построен конструктором.
  // При повторном использовании конструктора, внесенные
вручную изменения будут утеряны!!!
  Движения.ОстаткиМатериалов.Записывать = Истина;
  Для Каждого ТекСтрокаПереченьНоменклатуры Из
ПереченьНоменклатуры Цикл
    Если
ТекСтрокаПереченьНоменклатуры.Номенклатура.ВидНоменклатуры = Перечисления.ВидыНоменклатуры.Материал Тогда
      // регистр ОстаткиМатериалов Расход
      Движение = Движения.ОстаткиМатериалов.Добавить();
      Движение.ВидДвижения =
ВидДвиженияНакопления.Расход;
      Движение.Период = Дата;
      Движение.Материал =
ТекСтрокаПереченьНоменклатуры.Номенклатура;
      Движение.Склад = Склад;
      Движение.Количество =
ТекСтрокаПереченьНоменклатуры.Количество;
      КонецЕсли;
    КонецЦикла;
  //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Добавленный текст исключает из выполнения операторов цикла те строки документа, в которых номенклатура не является материалом. К значению перечисления «Материал» мы обращаемся, используя менеджер перечисления «ВидыНоменклатуры» (Перечисления.ВидыНоменклатуры), указывая в качестве его свойства имя нужного нам значения перечисления.

236. Выберите **Отладка**, **Начать отладку**, на появившийся вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне **Реорганизация информации** выберите **Принять**.

237. Откроем документ **Оказание услуги №1** и внесем в него следующие изменения: удалите из табличной части строку содержащую **Транзистор**, добавьте услугу **Подключение воды**, добавьте материал **Шлаг резиновый**, нажмите **Провести**.

Обратите внимание, что цены подставляются автоматически из регистра сведений **Цены**.

238. Выберите **Остатки материалов** в панели навигации, чтобы перейти к записям регистра **Остатки материалов**. Убедимся, что в движении по регистру «**ОстаткиМатериалов**» включаются только строки, содержащие материалы, т.е. шланг резиновый есть. А запись про услугу **Подключение воды** в движения не попала.

9-й день. Проведение документа по нескольким регистрам

9.1. Зачем нужно проведение документа по нескольким регистрам?

Мы учитывали только количественное движение материалов в ООО «На все руки мастер». Для этих целей мы создали регистр накопления «ОстаткиМатериалов». Однако, как вы, наверное, догадываетесь, одного только количественного учета совершенно недостаточно для нужд нашего ООО.

Очевидно, что необходимо также знать, какие денежные средства были затрачены на приобретение тех или иных материалов, и каковы материальные запасы ООО «На все руки мастер» в денежном выражении.

После того, как мы начали автоматизировать наше предприятие, руководство ООО «На все руки мастер» высказало пожелание, чтобы весь суммовой учет материалов велся бы теперь по средней стоимости. То есть, при закупке материалов они должны учитываться в ценах приобретения, а при расходе - по средней стоимости, которая рассчитывается исходя из общей суммы закупок данного материала и общего количества этого материала, находящегося в ООО.

Поскольку подобная информация имеет совершенно другую структуру, нежели количественный учет, для хранения данных об общей стоимости тех или иных материалов мы будем использовать еще один регистр накопления «СтоимостьМатериалов».

Таким образом, документы «ПриходнаяНакладная» и «Оказание-Услуги» должны будут создавать движения не только в регистре «ОстаткиМатериалов», но, одновременно, и в регистре «СтоимостьМатериалов», отражая изменения суммового учета.

9.2. Добавление регистра накопления

239. Создайте регистр: выберите в дереве объектов конфигурации ветвь **Регистр накопления, МП**, выберите **Добавить**, в поле **Имя** введите **СтоимостьМатериалов**, нажмите **tab** и в поле **Синоним** должно появиться **Стоимость материалов**, поле **Расширенное представление списка** введите **Движения по регистру Стоимость материалов**, выберите **Далее**.

240. На вкладке подсистемы выберите **Учет материалов, Оказание услуг и Бухгалтерия**.

241. Выберите вкладку **Данные**, выделите ветвь **Измерения**, выберите **Добавить**, в поле **Имя** введите **Материал**, в списке **Тип** выберите **СправочникСсылка.Номенклатура**.

242. Создайте ресурс **Стоимость** с длиной **15** и точностью **2**.

243. Сделайте доступной в панели действий разделов команду для просмотра записей регистра накопления: выберите в дереве объектов конфигурации ветвь **Подсистемы, МП, Все подсистемы**, в окне **Все подсистемы** слева в списке **Подсистемы** выберите подсистему **Бухгалтерия**, в группе **Панель навигации.Обычное** для команды **Стоимость материалов** включите видимость и перетащите ее в группу **Панель навигации. См. Также**.

243. Аналогично выполните для подсистем **Оказание услуг** и **УчетМатериалов**

9.3. Изменение процедуры проведения документа

244. Откроем в конфигураторе окно редактирования объекта конфигурации документ **ПриходнаяНакладная** и перейдем на закладку **Движения**. В списке регистров отметим, что документ будет создавать теперь движения и по регистру **СтоимостьМатериалов**.

Сейчас использование конструктора движений нецелесообразно. Так как при использовании конструктора пришлось бы заново описывать движения для обоих регистров. Поэтому проще внести изменения вручную.

245. Выберите **Прочее, Модуль объекта**.

246. Введите код (жирный):

```
Процедура ОбработкаПроведения(Отказ, Режим)
  {{{_КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
  // Данный фрагмент построен конструктором.
  // При повторном использовании конструктора, внесенные
вручную изменения будут утеряны!!!
  Движения.ОстаткиМатериалов.Записывать = Истина;
  Движения.СтоимостьМатериалов.Записывать = Истина;
  Для Каждого ТекСтрокаМатериалы Из Материалы Цикл
    // регистр ОстаткиМатериалов Приход
    Движение = Движения.ОстаткиМатериалов.Добавить();
```

```

        Движение.ВидДвижения =
ВидДвиженияНакопления.Приход;
        Движение.Период = Дата;
        Движение.Материал = ТекСтрокаМатериалы.Материал;
        Движение.Склад = Склад;
        Движение.Количество = ТекСтрокаМатериалы.Количество;

        Движение =
Движения.СтоимостьМатериалов.Добавить());
        Движение.ВидДвижения =
ВидДвиженияНакопления.Приход;
        Движение.Период = Дата;
        Движение.Материал =
ТекСтрокаМатериалы.Материал;
        Движение.Стоимость = ТекСтрокаМатериалы.Сумма;

        КонецЦикла;
    //}}_КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

```

247. Изменим командный интерфейс формы документа, чтобы в панели навигации формы иметь возможность переходить к списку записей регистра **СтоимостьМатериалов**: откройте форму документа **ПриходнаяНакладная**, выберите вкладку **Командный интерфейс**, раскройте группу **Перейти**, для команды открытия регистра накопления **Стоимость материалов** установите видимость.

248. Выберите **Отладка, Начать отладку**, на вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

249. Откройте документ: выберите раздел **Учет материалов**, выберите все документы **ПриходнаяНакладная**, в списке **Все действия** выберите **Провести**.

250. Откройте первый документ **Приходная накладная**.

251. Откройте регистры: выберите **Остатки материалов** и **Стоимость материалов**, разместите оба окна рядом.

Убедимся, что документ создает желаемые записи в обоих регистрах

9.4. Изменение процедуры проведения документа

На данном этапе мы будем исходить из пожелания, выданного руководством ООО «На все руки мастер». Суть его заключается в том, что на первом этапе, при списании материалов, израсходованных в процессе оказания услуги, должна быть возможность указывать различную стоимость для одного и того же материала, которая рассчитана руководством исходя из текущих конъюнктурных соображений.

Поскольку в документе «ОказаниеУслуги» у нас отражена только цена номенклатуры, нам понадобится добавить в табличную часть документа еще одно поле, в котором будет указываться стоимость номенклатуры.

252. Откроем в конфигураторе окно редактирования объекта конфигурации документ **ОказаниеУслуги**, перейдем на закладку **Данные** и создадим новый реквизит **табличной** части документа с именем **Стоимость**, типом **Число**, длиной **15** и точностью **2**.

253. После этого откроем форму **ФормаДокумента** документа **ОказаниеУслуги**, в правом верхнем окне на вкладке **Реквизиты** раскроем реквизит формы **Объект**, выберите реквизит **Стоимость**, с помощью мыши перетащите его в окно элементов формы, расположенное **слева** в верхней части редактора форм, расположите его после поля **Номенклатура**.

253. Откроем в конфигураторе окно редактирования объекта конфигурации документ **ОказаниеУслуги** и укажем, что он будет создавать движения по регистру накопления **СтоимостьМатериалов**.

254. Выберите **Прочее, Модуль объекта**. Опишем движения документа следующим образом (обратите внимание, что стоимость вычисляется как произведение стоимости и количества, указанных в табличной части):

```
Процедура ОбработкаПроведения(Отказ, Режим)
//{{_КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
// Данный фрагмент построен конструктором.
// При повторном использовании конструктора, внесенные
вручную изменения будут утеряны!!!
```

```

Движения.ОстаткиМатериалов.Записывать = Истина;
Движения.СтоимостьМатериалов.Записывать = Истина;
Для Каждого ТекСтрокаПереченьНоменклатуры Из
ПереченьНоменклатуры Цикл
    Если
ТекСтрокаПереченьНоменклатуры.Номенклатура.ВидНоменклатуры
= Перечисления.ВидыНоменклатуры.Материал Тогда
        // регистр ОстаткиМатериалов Расход
        Движение = Движения.ОстаткиМатериалов.Добавить();
        Движение.ВидДвижения =
ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Материал =
ТекСтрокаПереченьНоменклатуры.Номенклатура;
        Движение.Склад = Склад;
        Движение.Количество =
ТекСтрокаПереченьНоменклатуры.Количество;

        Движение =
Движения.СтоимостьМатериалов.Добавить();
        Движение.ВидДвижения =
ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Материал =
ТекСтрокаПереченьНоменклатуры.Номенклатура;
        Движение.Стоимость =
ТекСтрокаПереченьНоменклатуры.Количество*ТекСтрокаПере
ченьНоменклатуры.Стоимость;

        КонецЕсли;
    КонецЦикла;
//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

```

255. Изменим командный интерфейс формы документа, чтобы в панели навигации формы иметь возможность переходить к списку записей регистра СтоимостьМатериалов: откройте форму документа

Оказание Услуги, выберите вкладку **Командный интерфейс**, раскройте группу **Перейти**, для команды открытия регистра накопления **Стоимость материалов** установите видимость.

256. Выберите **Отладка, Начать отладку**, на вопрос: **редактируемая конфигурация отличается от конфигурации базы данных. Обновить конфигурацию базы данных?** выберите **Да**, в окне Реорганизация информации выберите **Принять**.

257. Откройте документ: выберите раздел **Оказание услуг**, выберите команду **Оказание услуг**, откройте документ **Оказание услуги**, введите стоимость резинового шланга 100 руб., нажмите **Провести**, выберите просмотр движений по регистру нажмите **Стоимость материалов**.

258. Создайте документ Оказание услуги 2 от 29 июля по рис.

N	Номенклатура	Стоимость	Количество	Цена	Су
1	Ремонт импортного теле...		1,000	800,00	
2	Строчный трансформатор...	600,00	1,000	900,00	

259. Нажмите **Провести**, нажмите **Стоимость материалов**. Движения должны быть как на рис.

Период	Регистратор	Номер строки	Материал	Склад
29.07.2011 12:00:00	Оказание услуги 000...	1	Строчный трансфор...	Основной

260. Создайте документ Оказание услуги 3 от 29 июля по рис.

Оказание услуги 000000003 от 29.07.2011 12:00:01

Провести и закрыть Провести Печать

Номер: 000000003

Дата: 29.07.2011 12:00:01

Склад: Основной

Клиент: Зырянов Алексей Владимирович

Мастер: Рогов Денис Владимирович

Добавить

N	Номенклатура	Стоимость	Количество	Цена
1	Подключение электричества		1,000	800,00
2	Шланг резиновый	100,00	2,000	150,00
3	Кабель электрический	20,00	1,000	30,00
4	Ремонт отечественного телевизора		1,000	600,00
5	Строчный трансформатор GoldStar	270,00	1,000	400,00
6	Транзистор Philips 2N2369	3,00	2,000	7,00

261. Нажмите Провести, нажмите Стоимость материалов. Движения должны быть как на рис.

Движения по регистру Стоимость материалов

Найти...

Период	Регистратор	Номер строки	Материал
29.07.2011 12:00:01	Оказание услуги 000000003 от 29.07.2011 12:00:01	1	Шланг резиновый
29.07.2011 12:00:01	Оказание услуги 000000003 от 29.07.2011 12:00:01	2	Кабель электрический
29.07.2011 12:00:01	Оказание услуги 000000003 от 29.07.2011 12:00:01	3	Строчный трансформатор GoldStar
29.07.2011 12:00:01	Оказание услуги 000000003 от 29.07.2011 12:00:01	4	Транзистор Philips 2N2369

10. Оборотные регистры накопления

10.1 Зачем нужно создавать еще один регистр

Продолжим рассматривать работу нашего документа «Оказание Услуги». До сих пор мы создавали в регистрах накопления движения только для строк документа, которые содержат материалы. Услуги, содержащиеся в документе, мы никак не учитывали.

Дело в том, что при учете услуг важны совершенно другие критерии, нежели при учете материалов. Прежде всего, бессмысленно говорить о том, сколько услуг было и сколько их осталось, важна только сумма и количество услуг, которые были оказаны за определенный промежуток времени. Кроме этого интересны следующие моменты:

- какие именно услуги были оказаны (чтобы составить рейтинг услуг)
- какому именно клиенту оказывались услуги (чтобы предоставить ему скидку от объема оплаченных ранее услуг ,
- какой мастер предоставлял услуги (чтобы начислить ему заработную плату)

Очевидно, что существующие регистры накопления совершенно не подходят для решения таких задач. Поэтому мы создадим еще одно «хранилище» данных, которое будет использоваться в нашей программе - оборотный регистр накопления «Продажи».

10.2. Что такое оборотный регистр накопления

Регистры накопления могут быть регистрами остатков и регистрами оборотов.

Существующие в нашей учебной конфигурации регистры «ОстаткиМатериалов» и «СтоимостьМатериалов» являются регистрами остатков. Если вы вспомните момент, когда мы создавали отчет «Материалы», то в конструкторе отчета мы видели, что для таких регистров система создает три виртуальные таблицы: таблица остатков, оборотов и совокупная таблица остатков и оборотов.

Оборотный регистр накопления очень похож на, знакомый уже нам, регистр остатков, для которого понятие «остаток» не имеет смысла. Оборотный регистр накапливает только обороты, остатки ему без-

различны. Поэтому единственной виртуальной таблицей, которую будет создавать система для такого регистра, будет таблица оборотов.

В остальном оборотный регистр ни чем не отличается от регистра остатков.

Следует сказать об одной особенности конструирования регистров накопления, напрямую связанной с возможностью получения остатков.

При создании оборотного регистра накопления нет особой сложности в определении того, какие именно параметры должны являться измерениями регистра - мы можем назначить в качестве его измерений любые нужные нам параметры.

Совсем иная ситуация в случае регистра накопления поддерживающего накопление остатков. Для него выбор измерений должен выполняться исходя из того, что движения регистра могут быть осуществлены «в две стороны»: приход и расход. Таким образом, в качестве измерений нужно выбирать те параметры, по которым движения точно будут осуществляться как в одну, так и в другую сторону.

Например, если ведется учет материалов в разрезах номенклатуры и склада - очевидно, что и номенклатура и склад могут быть измерениями, поскольку как приход, так и расход материалов всегда будет осуществляться с указанием конкретной номенклатуры и конкретного склада. Если же в этой ситуации появляется желание отразить учет материалов еще и в разрезе поставщика, то здесь уже нужно исходить из конкретной схемы учета, принятой на предприятии.

Скорее всего, при поступлении материалов поставщик будет указан, а вот при расходе материалов, с большой долей вероятности поставщик указываться не будет, так как в большинстве случаев это совершенно лишняя информация. Значит, поставщика следует добавить как реквизит регистра накопления.

Если же при расходе материалов поставщик будет указываться наверняка, тогда имеет смысл добавить поставщика в измерения регистра.

Иными словами, по каждому из измерений регистра накопления остатков изменение ресурсов обязательно должно осуществляться в обе стороны: приход и расход.

Для реквизитов регистра этот принцип неважен, по реквизитам регистра ресурсы могут только приходоваться или только расходоваться.

Нарушение этого принципа построения регистров накопления будет вести к непроизводительному использованию ресурсов системы и, как следствие, замедлению работы и падению производительности.

10.3. Создание оборотного регистра накопления

Теперь, когда мы знаем «практически все» о регистрах накопления, откроем конфигуратор и создадим новый объект конфигурации регистр накопления. Назовем его «Продажи» и определим вид регистра - «Обороты».

262. Создадим новый объект конфигурации Регистр накопления: выберите на дереве объект **Регистр накопления, МП**, выберите **Добавить**, в поле **Имя** введите **Продажи**, в поле **Расширенное представление списка** введите **Движения по регистру Продажи**, нажмите **Далее**.

263. На вкладке **Подсистемы** выберите **Бухгалтерия, Учет материалов, Оказание услуг**.

264. На закладке **Данные** создадим измерения регистра:

Номенклатура, тип СправочникСсылка.Номенклатура,

Клиент, тип СправочникСсылка.Клиенты,

Мастер, тип СправочникСсылка.Сотрудники.

265. Создайте у регистра три ресурса:

Количество, тип Число, длина 20, точность 3,

Выручка, тип Число, длина 15, точность 2,

Стоимость, тип Число, длина 15, точность 2.

266. Сделайте доступной в панели действий разделов команду для просмотра записей регистра накопления: выберите в дереве объектов конфигурации выделите ветвь **Подсистемы, МП, Все подсистемы**, в окне **Все подсистемы** слева в списке **Подсистемы** выберите подсистему **Бухгалтерия**, в группе **Панель навигации.Обычное** для команды **Продажи** включите видимость и перетащите ее в группу **Панель навигации**. См. Также.

267. Аналогично выполните для подсистем **Оказание услуг** и **УчетМатериалов**

268. Откроем окно редактирования объекта конфигурации Документ **ОказаниеУслуги** и на закладке **Движения** укажем, что этот документ будет создавать движения по регистру **Продажи**.

269. Выберите вкладку **Прочее, Модуль объекта.**

270. Введите код создающий движения регистра Продажи, производимые документом ОказаниеУслуги, выделенный жирным:

```
Процедура ОбработкаПроведения(Отказ, Режим)
  //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
  // Данный фрагмент построен конструктором.
  // При повторном использовании конструктора, внесенные
вручную изменения будут утеряны!!!
  Движения.ОстаткиМатериалов.Записывать = Истина;
  Движения.СтоимостьМатериалов.Записывать = Истина;
  Движения.Продажи.Записывать = Истина;

  Для Каждого ТекСтрокаПереченьНоменклатуры Из
ПереченьНоменклатуры Цикл
    Если
ТекСтрокаПереченьНоменклатуры.Номенклатура.ВидНоменклатуры
= Перечисления.ВидыНоменклатуры.Материал Тогда
      // регистр ОстаткиМатериалов Расход
      Движение = Движения.ОстаткиМатериалов.Добавить();
      Движение.ВидДвижения =
ВидДвиженияНакопления.Расход;
      Движение.Период = Дата;
      Движение.Материал =
ТекСтрокаПереченьНоменклатуры.Номенклатура;
      Движение.Склад = Склад;
      Движение.Количество =
ТекСтрокаПереченьНоменклатуры.Количество;

      Движение = Движения.СтоимостьМатериалов.Добавить();
      Движение.ВидДвижения =
ВидДвиженияНакопления.Расход;
      Движение.Период = Дата;
      Движение.Материал =
ТекСтрокаПереченьНоменклатуры.Номенклатура;
```

```
Движение.Стоимость =
ТекСтрокаПереченьНоменклатуры.Количество*ТекСтрокаПереченьНоменклатуры.Стоимость;
```

```
КонецЕсли;
```

```
Движение = Движения.Продажи.Добавить();
```

```
Движение.Период = Дата;
```

```
Движение.Номенклатура =
```

```
ТекСтрокаПереченьНоменклатуры.Номенклатура;
```

```
Движение.Клиент = Клиент;
```

```
Движение.Мастер = Мастер;
```

```
Движение.Количество =
```

```
ТекСтрокаПереченьНоменклатуры.Количество;
```

```
Движение.Выручка =
```

```
ТекСтрокаПереченьНоменклатуры.Сумма;
```

```
Движение.Стоимость =
```

```
ТекСтрокаПереченьНоменклатуры.Стоимость*ТекСтрокаПереченьНоменклатуры.Количество;
```

```
КонецЦикла;
```

```
//}}_КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
```

```
КонецПроцедуры
```

271. Запустим 1С:Предприятие в режиме отладки, откройте документ ОказаниеУслуги от 27 июля, нажмите Провести, перейдите к списку движений по регистру Продажи.

Движения по регистру Продажи,

(++) | Найти... | Все действия ▾

Номер строки	Номенклатура	Клиент	Мастер	Количество	Выручка	Стоимость
1	Подключение воды	Арляпов Антон Андр...	Пинчук Денис Алекс...	1,000	800,00	
2	Шланг резиновый	Арляпов Антон Андр...	Пинчук Денис Алекс...	1,000	150,00	100,00

272. Откройте документ ОказаниеУслуги от 29 июля, нажмите Провести, перейдите к списку движений по регистру Продажи.

Движения по регистру Продажи,

(++) | Найти... | Все действия ▾

Номер строки	Номенклатура	Клиент	Мастер	Количество	Выручка	Стоимость
1	Подключение воды	Арляпов Антон Андр...	Пинчук Денис Алекс...	1,000	800,00	
2	Шланг резиновый	Арляпов Антон Андр...	Пинчук Денис Алекс...	1,000	150,00	100,00
1	Ремонт импортного ...	Василихина Ольга Г...	Голендова Марина Е...	1,000	800,00	
2	Строчный трансфор...	Василихина Ольга Г...	Голендова Марина Е...	1,000	900,00	600,00

273. Откройте документ ОказаниеУслуги от 29 июля, нажмите Провести, перейдите к списку движений по регистру Продажи.

Движения по регистру Продажи,

(←) | 🔍 Найти... | 🔄 Все действия ▾

Номер строки	Номенклатура	Клиент	Мастер	Количество	Выручка	Стоимость
1	Подключение воды	Арляпов Антон Андр...	Пинчук Денис Алекс...	1,000	800,00	
2	Шланг резиновый	Арляпов Антон Андр...	Пинчук Денис Алекс...	1,000	150,00	100,00
1	Ремонт импортного ...	Василихина Ольга Г...	Голендова Марина Е...	1,000	800,00	
2	Строчный трансфор...	Василихина Ольга Г...	Голендова Марина Е...	1,000	900,00	600,00
1	Подключение электр...	Зырянов Алексей Вл...	Рогов Денис Владим...	1,000	800,00	
2	Шланг резиновый	Зырянов Алексей Вл...	Рогов Денис Владим...	2,000	300,00	200,00
3	Кабель электрический	Зырянов Алексей Вл...	Рогов Денис Владим...	1,000	30,00	20,00
4	Ремонт отечественн...	Зырянов Алексей Вл...	Рогов Денис Владим...	1,000	600,00	
5	Строчный трансфор...	Зырянов Алексей Вл...	Рогов Денис Владим...	1,000	400,00	270,00
6	Транзистор Philips 2...	Зырянов Алексей Вл...	Рогов Денис Владим...	2,000	14,00	6,00

11-й день. Отчеты

11.1. Способы доступа к данным

Система 1С: предприятие поддерживает два способа доступа к данным, хранящихся в БД:

- объектный (для чтения и записи)
- табличный (для чтения).

Объектный способ доступа к данным реализован посредством использования объектов встроенного языка. При этом обращаясь к какому-либо объекту встроенного языка, мы обращаемся к некоторой совокупности данных, находящихся в БД, как к **единому** объекту.

Например, объект ДокументОбъект.ОказаниеУслуги будет содержать значения всех реквизитов документа Оказание услуги и всех его табличных частей.

Табличный доступ к данным реализован с помощью запросов к БД, которые составляются на языке запросов. Здесь разработчик получает возможность оперировать отдельными полями таблиц БД, к которых хранятся те или иные данные.

11.2. Работа с запросами

Для формирования и выполнения запросов к таблицам базы данных в системе используется специальный объект Запрос. Запрос удобно использовать, когда необходимо получить сложную выборку данных, сгруппированную и отсортированную нужным образом. Одним из классических примеров его применения может служить сводка по состоянию регистра учета на определенный момент времени. Кроме того, механизм запросов позволяет легко получать информацию в различных временных разрезах.

11.3. Источники данных запросов

Исходную информацию запрос получает из набора таблиц.

Все таблицы, которыми оперирует язык запросов можно разделить на две группы:

- реальные таблицы
- виртуальные таблицы.

Реальные таблицы содержат данные какой-либо одной реальной таблицы, хранящейся в БД.

Например, реальной является таблица Справочник.Клиенты соответствующая справочнику Клиенты.

Виртуальные таблицы формируются в основном из данных нескольких таблиц БД. Например, виртуальной является таблица РегистрНакопления.ОстаткиМатериалов.ОстаткиИОбороты, формируемая из нескольких таблиц регистра накопления Остатки материалов.

Общим для них является то, что им можно задать ряд параметров, определяющих какие данные будут включены в эти виртуальные таблицы.

Виртуальны таблицы не хранятся в БД.

Реальные таблицы делятся на объектные (ссылочные) и неobjектные.

В объектных таблицах представлена информация ссылочных типов данных (справочники, документы, и т.д.). А в неobjектных – всех остальных типов данных (константы, регистры и т.д.).

Особенностью объектных таблиц является то, что они включают в себя поле Ссылка, содержащее ссылку на текущую запись.

11.4. Язык запросов

Алгоритм по которому данные будут выбраны из исходных таблиц запроса, описывается на специальном языке – языке запросов.

Текст запроса может состоять из частей:

1. описание запроса
2. объединение запросов
3. упорядочивание результатов
4. автоупорядочивание
5. описание итогов.

Обязательной частью является только – описание запроса.

Описание запроса – определяет источники данных, поля выборки, группировки и т.д.

Объединение запросов – определяет как будут объединены результаты выполнения нескольких запросов.

Упорядочивание результатов – определяет условия упорядочивания строк результата запроса.

Автоупорядочивание позволяет включать режим автоматического упорядочивания строк результата запроса.

Описание итогов – определяет, какие итоги необходимо рассчитать в запросе и каким образом группировать результат.

Система компоновки данных



Система компоновки данных предназначена для создания произвольных отчетов.

Исходные данные для компоновки отчета содержит в себе схема компоновки данных. Это наборы данных и методы работы с ними.

Разработчик создает схему компоновки данных, в которой описывает текст запроса, наборы данных, связи между ними, доступные поля, параметры получения данных, и задает первоначальные настройки компоновки – структуру отчета, макет оформления данных и пр.



Разработчик создает схему компоновки данных и настройки по умолчанию.

На основе схемы и настроек компоновщик макета создает макет.

Процессор компоновки данных выбирает данные из ИБ согласно макету компоновки, агрегирует и оформляет эти данные.

Результат компоновки обрабатывается процессором вывода, и в итоге пользователь получает результирующий табличный документ.

11.5. Выбор данных из одной таблицы

274. Создадим отчет: выберите в дереве объектов конфигурации ветвь **Отчеты**, **МП**, выберите **Добавить**, в поле **Имя** введите **РеестрДокументовОказаниеУслуги**, нажмите **tab** и в поле **Синоним** должно появиться **РеестрДокументовОказаниеУслуги**, в поле расширенное представление введите **Список оказанных услуг**,

275. нажмите **Открыть схему компоновки данных**. В окне конструктора макета выберите тип макета **Схема компоновки данных**, нажмите **Готово**.

276. Добавьте новый набор данных – запрос: нажмите кнопку **Добавить**, выберите **Добавить набор данных – запрос**.

277. Создайте текст запроса: нажмите кнопку **Конструктор запроса**.

278. В качестве источника данных для запроса выберем объектную таблицу **ОказаниеУслуги**, из этой таблицы выберем поля **Склад, Мастер, Клиент, Ссылка**.

279. Выберите вкладку **Объединения/Псевдонимы**, укажите что поле **Ссылка** будет иметь псевдоним **Документ**.

280. Выберите вкладку **Порядок**, укажите что результат запроса должен упорядочен по значению поля **Документ**.

281. Нажмите ОК.

ВЫБРАТЬ

ОказаниеУслуги.Склад,

ОказаниеУслуги.Мастер,

ОказаниеУслуги.Клиент,

ОказаниеУслуги.Ссылка **КАК** Документ

ИЗ

Документ.ОказаниеУслуги **КАК** ОказаниеУслуги

УПОРЯДОЧИТЬ ПО

Документ

ВЫБРАТЬ

ОказаниеУслуги.Склад,
ОказаниеУслуги.Мастер,
ОказаниеУслуги.Клиент,
ОказаниеУслуги.Ссылка **КАК** Документ

ИЗ

Документ.ОказаниеУслуги **КАК** ОказаниеУслуги

УПОРЯДОЧИТЬ ПО

Документ

Описание запроса

Упорядочивание результата (по умолчанию по возрастанию)

Список полей выборки

После **КАК** псевдоним источника данных

Источники данных

282. выберите вкладку **настройки**, выберите **Отчет, МП, Новая группировка**.

В структуре отчета появится группировка детальные записи.

283. На вкладке **Выбранные поля** перенесем мышью поля.

284. Закроем конструктор схемы компоновки данных.

285. В окне редактирования объекта конфигурации **Отчет РеестрДокументовОказаниеУслуг** выберите вкладку **Подсистемы**, выберите **Оказание услуг**.

286. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела **Оказание услуг** выберите **Реестр документов оказания услуг**, нажмите **Сформировать**.

Мы видим, что отчет содержит реестр документов **Оказание услуг**. Причем двойным щелчком на поле **Документ** мы можем открыть исходный документ, а также выполнить другие действия.

11.6. Выбор данных из двух таблиц

Отчет Рейтинг услуг будет содержать информацию о том, выполнение каких услуг принесло ООО наибольшую прибыль в указанном периоде.

286. Создадим отчет: выберите в дереве объектов конфигурации ветвь **Отчеты, МП**, выберите **Добавить**, в поле **Имя** введите **РейтингУслуги**, нажмите **tab** и в поле **Синоним** должно появиться **РейтингУслуги**.

287. нажмите **Открыть схему компоновки данных**. В окне конструктора макета выберите тип макета **Схема компоновки данных**, нажмите **Готово**.

288. Добавьте новый набор данных – запрос: нажмите кнопку **Добавить**, выберите **Добавить набор данных – запрос**.

289. Создайте текст запроса: нажмите кнопку **Конструктор запроса**.

290. В качестве источника данных для запроса выберем объектную таблицу **Номенклатура** и виртуальную таблицу регистра накопления **ПродажиОбороты**.

290. Переименуем таблицу **Номенклатура** в **спрНоменклатура**

291. В список полей перенесем поля **СпрНоменклатура.Ссылка** и **ПродажиОбороты.ВыручкаОборот**.

292. выберите вкладку **связи**. Так как в запросе участвуют несколько таблиц требуется определить связь между ними.

По умолчанию платформой уже создана связь по полю **Номенклатура**. То есть значения измерения **Номенклатура** регистра **Продажи** должно быть равно ссылке на элемент справочника **Номенклатура**.

293. Снять флажок **Все** у у таблицы **ПродажиОбороты** и установить у таблицы **спрНоменклатура**.

Это будет тип связи **левое соединение**, то есть в результат запроса будут включены все записи справочника **Номенклатура** и те записи регистра **Продажи**, которые удовлетворяют условию связи по полю **Номенклатура**.

Ссылка	Номенклатура	Выручка (Оборот)
Материалы		
Услуги		
Строчный трансформатор Samsung	Строчный трансформатор Samsung	900,00
Строчный трансформатор GoldStar	Строчный трансформатор GoldStar	400,00
Транзистор Philips 2N2369	Транзистор Philips 2N2369	14,00
Шланг резиновый	Шланг резиновый	450,00
Кабель электрический	Кабель электрический	30,00
Диагностика		
Ремонт отечественного телевизора	Ремонт отечественного телевизора	600,00
Ремонт импортного телевизора	Ремонт импортного телевизора	800,00
Подключение воды	Подключение воды	800,00
Подключение электричества	Подключение электричества	800,00
Телевизоры		
Стиральные машины		
Радиодетали		
Прочее		

Ссылка	Выручка (Оборот)
Материалы	
Услуги	
Строчный трансформатор Samsung	900,00
Строчный трансформатор GoldStar	400,00
Транзистор Philips 2N2369	14,00
Шланг резиновый	450,00
Кабель электрический	30,00
Диагностика	
Ремонт отечественного телевизора	600,00
Ремонт импортного телевизора	800,00
Подключение воды	800,00
Подключение электричества	800,00
Телевизоры	
Стиральные машины	
Радиодетали	
Прочее	

В результате запроса будут присутствовать все услуги, и для некоторых из них будут указаны обороты выручки.

294. Выберите вкладку Условия и установим отбор, чтобы группы справочника Номенклатура не попадали в отчет.

295. Выберите спрНоменклатура, выберите поле ЭтаГруппа, установите флажок Произвольное, в поле Условие введите код:

спрНоменклатура.ЭтаГруппа = ЛОЖЬ

Ссылка	Выручка (Оборот)	Ссылка	Выручка (Оборот)
Материалы		Строчный трансформатор Samsung	900,00
Услуги		Строчный трансформатор GoldStar	400,00
Строчный трансформатор Samsung	900,00	Транзистор Philips 2N2369	14,00
Строчный трансформатор GoldStar	400,00	Шланг резиновый	450,00
Транзистор Philips 2N2369	14,00	Кабель электрический	30,00
Шланг резиновый	450,00	Диагностика	
Кабель электрический	30,00	Ремонт отечественного телевизора	600,00
Диагностика		Ремонт импортного телевизора	800,00
Ремонт отечественного телевизора	600,00	Подключение воды	800,00
Ремонт импортного телевизора	800,00	Подключение электричества	800,00
Подключение воды	800,00	Телевизоры	
Подключение электричества	800,00	Стиральные машины	
Телевизоры		Радиодетали	
Стиральные машины		Прочее	
Радиодетали			
Прочее			

296. Другое условие то, что выбранный элемент является услугой. Это Простое условие. Перетащите поле ВидНоменклатуры в список условий. Платформа автоматически сформирует условие, согласно которому вид номенклатуры должен быть равен значению параметра ВидНоменклатуры. Дальше перед выполнением запроса мы передадим в параметр ВидНоменклатуры значение перечисления – Услуга.

Ссылка	Выручка (Оборот)
Строчный трансформатор Samsung	900,00
Строчный трансформатор GoldStar	400,00
Трансistor Philips 2N2369	14,00
Шланг резиновый	450,00
Кабель электрический	30,00
Диагностика	
Ремонт отечественного телевизора	600,00
Ремонт импортного телевизора	800,00
Подключение воды	800,00
Подключение электричества	800,00

Ссылка	Выручка (Оборот)
Диагностика	
Ремонт отечественного телевизора	600,00
Ремонт импортного телевизора	800,00
Подключение воды	800,00
Подключение электричества	800,00

297. Выберите вкладку Объединение/Псевдонимы, поле Ссылка будет иметь псевдоним **Услуга**, а поле регистра **Выручка**.

298. Выберите вкладку **Порядок**, выберите **Выручка**, укажите что результат запроса должен отсортирован по убыванию значения поля выручка.

299. Нажмите ОК.

ВЫБРАТЬ

спрНоменклатура.Ссылка КАК Услуга,

ПродажиОбороты.ВыручкаОборот КАК Выручка

ИЗ

Справочник.Номенклатура КАК спрНоменклатура

ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.Продажи.Обороты КАК ПродажиОбороты

ПО ПродажиОбороты.Номенклатура = спрНоменклатура.Ссылка

ГДЕ

спрНоменклатура.ЭтоГруппа = ЛОЖЬ

И спрНоменклатура.ВидНоменклатуры = &ВидНоменклатуры

УПОРЯДОЧИТЬ ПО

Выручка УБЫВ

Ресурсы

Под ресурсами в системе компоновки данных подразумеваются поля, значения которых рассчитываются на основании детальных записей, входящих в группировку. Ресурсы являются групповыми или общими итогами отчета.

300. Выберите вкладку **Ресурсы**, выберите **Выручка**, выберите >>, чтобы конструктор выбрал все доступные ресурсы, по которым можно вычислять итог. У нас это ресурс Выручка.

Параметры

Пользователя интересуют данные о хозяйственной деятельности за определенный период. Поэтому в любом отчете есть параметры, задающие начало и конец отчетного периода.

Параметры отчета задают условия отбора записей в отчет.

301. Выберите вкладку **Параметры**

302. Избавим пользователя от необходимости указывать время при вводе даты периода, за который формируется отчет: выберите в строке НачалоПериода поле Дата, M2, в списке Состав даты выберите Дата, нажмите ОК.

303. Для параметра КонецПериода установите флажок Ограничение доступности.

304. нажмите кнопку Добавить, в поле Имя введите ДатаОкончания, в списке Тип выберите Дата, укажите Состав даты – Дата.

305. Выберите параметр КонецПериода, в поле Выражение введите выражение

КонецПериода(&ДатаОкончания, "День")

306. Выберите строку **ВидНоменклатуры**, в списке столбца **Значение** выберите **Услуга**.

Настройки

307. выберите вкладку настройки, выберите Отчет, МП, Новая группировка.

В структуре отчета появится группировка детальные записи.

308. На вкладке Выбранные поля перенесем мышью поля Услуга, Выручка.

309. Выберите вкладку Другие настройки, введите заголовок отчета – Рейтинг услуг.

Быстрые пользовательские настройки

310. Выберите вкладку **Параметры**, выберите Дата начала, нажмите кнопку Свойства элемента пользовательских настроек, включите флажком Включать в пользовательские настройки, нажмите ОК.

311. Выберите вкладку Параметры, выберите Дата окончания, нажмите кнопку Свойства элемента пользовательских настроек, включите флажком Включать в пользовательские настройки, нажмите ОК.

312. Для поля Дата начала в списке Значения выберите Начало этого месяца.

313. Для поля Дата окончания в списке Значения выберите Начало этого дня.

314. Закроем конструктор схемы компоновки данных.

315. В окне редактирования объекта конфигурации Отчет Рейтинг Услуг выберите вкладку Подсистемы, выберите Оказание услуг.

316. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела Оказание услуг выберите Рейтинг услуг.

Условное обозначение

317. В конфигураторе откройте схему компоновки данных на вкладке **Настройки**, в нижней части окна выберите вкладку **Условное обозначение**, нажмите кнопку **Добавить**.

318. В поле **Оформление** выберите синий цвет текста, нажмите ОК.

319. Затем укажем **Условие**, при наступлении которого будет применяться оформление, выберите **Новый элемент**, нажмите кнопку **Добавить**, в графе **Левое** значение укажите **Выручка**, в графе **Вид сравнения** укажите **Меньше**, в графе **Правое** значение укажите **700**, нажмите ОК.

То есть когда в поле Выручка окажется значение меньше 700, что-то будет выделено красным цветом.

320. Теперь зададим список оформляемых полей: в поле **Оформляемые** поля нажмите три точки, нажмите **Добавить**, выберите **Услуга**, выберите **Выручка**, нажмите ОК.

321. В поле **Представление** условного обозначения введите **Непопулярная услуга**. Это то, что увидит пользователь в своих настройках.

322. Теперь добавим созданное условие в пользовательские настройки: нажмите кнопку **Свойства элементов пользовательских**

настроек, установите флажок **Включать в пользовательские настройки** и установите свойство **Режим редактирования** в значение **Обычный**.

Мы включили созданную нами настройку условного оформления в обычные пользовательские настройки. Эти настройки в отличии от быстрых настроек, расположены не в форме отчета, а вызываются нажатием кнопки **Настройка**.

323. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела **Оказание услуг** выберите **Рейтинг услуг**, нажмите **Сформировать**.

Мы видим, что суммы услуг менее 700 руб. выделены красным цветом.

323. Нажмите кнопку **Настройка**, снимите флажок с настройки **Непопулярная услуга**, нажмите **Завершить редактирование**.

324. Нажмите **сформировать**, видите выделение цветом исчезло.

Пользовательские настройки

325. В конфигураторе на вкладке **Настройки** схемы компоновки данных содержатся полные настройки отчета, которые задает разработчик. Часть из них может быть представлена пользователю для создания произвольного отбора, условного оформления отчета и др.

326. Нажмите кнопку **Свойства элемента пользовательских настроек**, расположенных вверху в командной панели окна настроек.

327. Установите признак использования для настроек **Отбор** и **Условное оформление** и установите для них режим редактирования в значение **Обычный**, нажмите **ОК**.

Отбор

328. Выберите вкладку **Отбор**, раскройте поле **Услуга**, выберите поле **Родитель**, М2, перенесите его в список отбора в правой части окна.

Мы создали возможность отбора по группам услуг, которые пользователь может задать в режиме 1С:Предприятие.

329. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела **Оказание услуг** выберите **Рейтинг услуг**, нажмите **Настройка**, там появились настройки **Отбор** и **Условное обозначение**.

Настройку **Непопулярная услуга** мы заранее создали в конфигураторе. А теперь, добавив настройку условного обозначения вообще,

мы представили пользователю возможность создавать любое количество собственных условий.

330. Зададим отбор в отчете так, чтобы в него попадали такие услуги, относящиеся у установке стиральных машин: нажмите три точки в окне пользовательских настроек в строке Отбор: в строке Отбор нажмите три точки, в строке Значение нажмите три точки, раскройте группу Услуги и выберите Стиральные машины из справочника Номенклатура, нажмите ОК, нажмите Завершить редактирование, нажмите Сформировать.

В отчет только услуги по установке стиральных машин.

331. Нажмите Настройка, в строке Отбор нажмите кнопку Очистка.

11.7. Отчет 3. Вывод данных по всем дням в выбранном периоде

Отчет Выручка мастеров будет содержать информацию о том, какая выручка была получена ООО благодаря работе мастеров, с детализацией по дням в выбранном периоде и разворотом по клиентам, обслуженным в каждый из дней.

331. Создадим отчет: выберите в дереве объектов конфигурации ветвь **Отчеты, МП**, выберите **Добавить**, в поле **Имя** введите **ВыручкаМастеров**, нажмите **tab** и в поле **Синоним** должно появиться **Выручка матеров**, в поле расширенное представление введите Список оказанных услуг,

332 нажмите **Открыть схему компоновки данных**. В окне конструктора макета выберите тип макета **Схема компоновки данных**, нажмите **Готово**.

333. Добавьте новый набор данных – запрос: нажмите кнопку **Добавить**, выберите **Добавить набор данных – запрос**.

334. Создайте текст запроса: нажмите кнопку **Конструктор запроса**.

335. В качестве источника данных для запроса выберем виртуальную таблицу регистра накопления **Продажи.Обороты**.

336. В поле Таблицы выберите **Продажи.Обороты**, нажмите кнопку **Парааметры виртуальной таблицы**, в списке Периодичность выберите **День**, нажмите **ОК**.

337. Выберите из таблицы поля **ПродажиОбороты.Мастер, ПродажиОбороты.Период, ПродажиОбороты.Клиент, ПродажиОбороты.ВыручкаОборот.**

338. Выберите вкладку **Объединения/Псевдонимы**, укажите что поле **ПродажиОбороты.ВыручкаОборот** будет иметь псевдоним **Выручка**, нажмите **ОК**.

ВЫБРАТЬ

**ПродажиОбороты.Мастер,
ПродажиОбороты.Период,
ПродажиОбороты.Клиент,
ПродажиОбороты.ВыручкаОборот КАК Выручка**

ИЗ

РегистрНакопления.Продажи.Обороты(, , День,) КАК ПродажиОбороты

Ресурсы

339. Выберите вкладку **Ресурсы**, выберите **выручка**.

Параметры

340. Выберите вкладку **Параметры**, для параметра **НачалоПериода** введите **Заголовк Дата начала**, в поле **Тип** выберите **состав даты Дата**.

342. Добавьте параметр **ДатаОкончания**, тип **Дата**, состав даты – **Дата**.

343. Для **КонецПериода** задайте выражение

КонецПериода(&ДатаОкончания, «День»)

и поле **Ограничение доступности** установите флажок.

Настройки

344. Выберите вкладку **Настройки**, выделите **корневой элемент Отчет**, нажмите **Добавить**, добавьте группировку **верхнего уровня** по полю **Мастер**, добавьте группировку **вложенную** в предыдущую по полю **Период**, добавьте еще одну группировку **вложенную** в группировку **Детальные записи** по полю **период** без указания группировочного поля.

345. Выберите вкладку **Выбранные поля**, добавьте поля **Клиент, Выручка**.

346. Выберите вкладку Другие настройки, в списке Расположение полей группировки выберите Отдельно и только в итогах, в списке Расположение общих итогов по вертикали выберите Начало, в поле Заголовок введите Выручка мастеров.

347. Выберите вкладку **Параметры**, выберите Дата начала, нажмите кнопку Свойства элемента пользовательских настроек, включите флажком Включать в пользовательские настройки, нажмите ОК.

348. Выберите вкладку Параметры, выберите Дата окончания, нажмите кнопку Свойства элемента пользовательских настроек, включите флажком Включать в пользовательские настройки, нажмите ОК.

349. В окне редактирования объекта конфигурации Отчет Выручка мастеров выберите вкладку Подсистемы, выберите Оказание услуг и Расчет зарплаты.

350. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела Оказание услуг выберите Выручка мастеров, задайте период с 1 июля по 30 июля, нажмите Сформировать.

Вывод всех дат в выбранном периоде

У нас показываются только те дни, для которых существуют ненулевые данные в таблице регистра накопления Продажи. Нам нужно показывать данные с детализацией по все дням в выбранном периоде.

351. В схеме компоновки данных выберите Настройки, выберите группировку Период, нажмите вкладку Период в командной панели окна.

352. Выберите вкладку Поля группировки, выберите поле Период, в списке Тип дополнения выберите день.

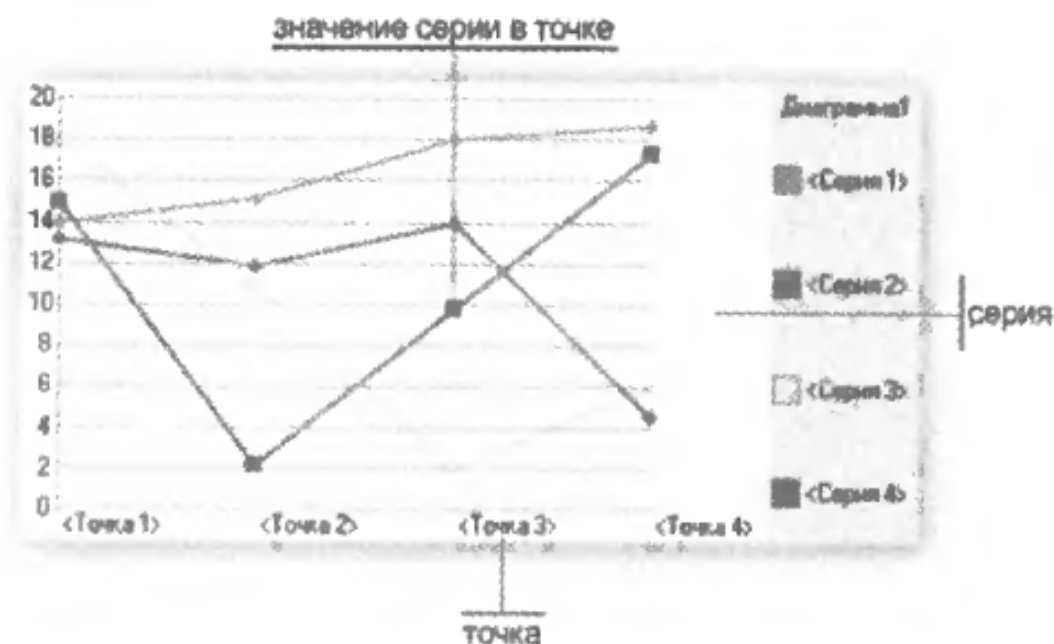
353. В новой строке в столбце Начальная дата периода, M2, нажмите кнопку Очистка, нажмите кнопку выбора типа данных T, выберите Поле компоновки данных, нажмите ОК, выберите три точки, выберите параметр НачалоПериода.

354. В новой строке в столбце Конечная дата периода, M2, нажмите кнопку Очистка, нажмите кнопку выбора типа данных T, выберите Поле компоновки данных, нажмите ОК, выберите три точки, выберите параметр ДатаОкончания.

355. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела Оказание услуг выберите Выручка мастеров, задайте период с 1 июля по 30 июля, нажмите Сформировать.

Новый вариант отчета. Диаграмма

Логически диаграмма является совокупностью точек, серий и значений серий в точке.



В качестве точек используются моменты или объекты для которых мы получаем значения характеристик, в качестве серий – характеристики, значения которых нас интересуют. На пересечении серии и точки находится значение диаграммы.

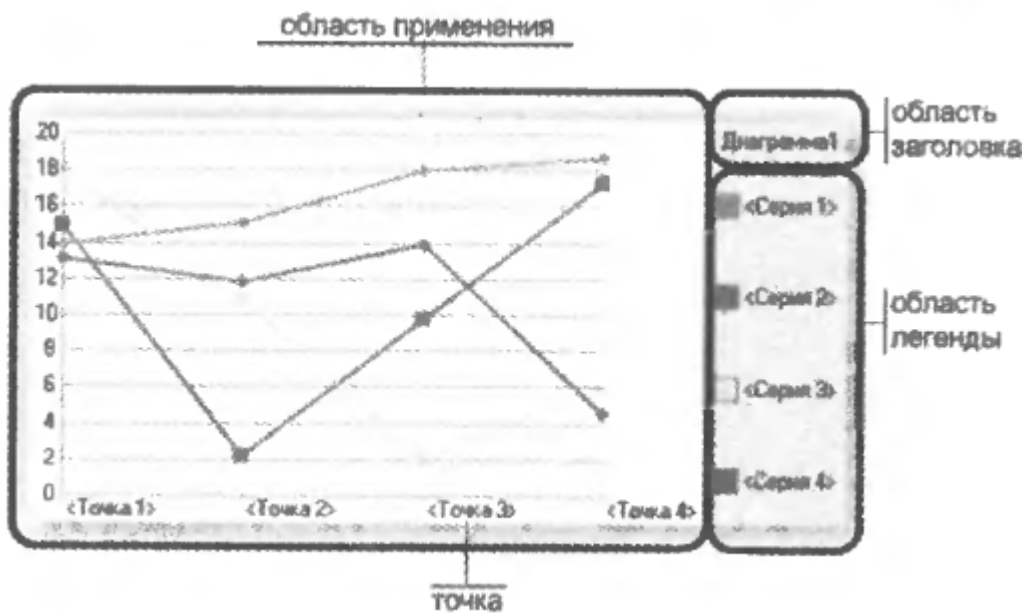
Например, диаграмма продаж видов номенклатуры по месяцам состоит из точек – месяцев, серий – видов номенклатуры и значений – оборотов продаж.

Диаграмма как объект встроенного языка имеет три области:

область построения

область заголовка

область легенды



356. Откройте схему компоновки данных на вкладке Настройки, в списке вариантов отчета выберите Добавить, введите имя ОбъемВыручки.

357. Добавим в структуру отчета диаграмму: выберите корневой элемент Отчет, МП, Новая диаграмма.

358. Выберите ветвь Точки, МП, Новая группировка, выберите поле Мастер.

359. Выберите Выбранные поля, нажмите Отчет, выберите Выручка.

360. Выберите вкладку Другие настройки, выберите Тип диаграммы – Измерительная.

361. Выберите полосы измерительной диаграммы по рис.

Полосы

Начало	Конец	Цвет	Текст
0	999	255, 255, 153	Плохо
1000	2000	50, 205, 50	Хорошо
2001	3000	255, 0, 255	Отлично

Закреть Справка

362. Выберите вкладку **Параметры**, выберите Дата начала, нажмите кнопку Свойства элемента пользовательских настроек, включите флажком Включать в пользовательские настройки, нажмите ОК.

363. Выберите вкладку **Параметры**, выберите Дата окончания, нажмите кнопку Свойства элемента пользовательских настроек, включите флажком Включать в пользовательские настройки, нажмите ОК.

364. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела Оказание услуг выберите Выручка мастеров, нажмите выбрать вариант, выберите Объем выручки, нажмите выбрать, задайте период с 1 июля по 30 июля, нажмите Сформировать.

11.8. Отчет 4. Получение актуальных значений из периодического регистра сведений

Отчет будет содержать информацию о том, какие услуги и по какой цене оказывает ООО.

365. Создадим отчет: выберите в дереве объектов конфигурации ветвь **Отчеты, МП**, выберите **Добавить**, в поле **Имя** введите **Перечень Услуг**, нажмите tab и в поле **Синоним** должно появиться **Перечень услуг**, в поле расширенное представление введите Список оказанных услуг,

366 нажмите **Открыть схему компоновки данных**. В окне конструктора макета выберите тип макета **Схема компоновки данных**, нажмите **Готово**.

367. Добавьте новый набор данных – запрос: нажмите кнопку **Добавить**, выберите **Добавить набор данных – запрос**.

368. Создайте текст запроса: нажмите кнопку **Конструктор запроса**.

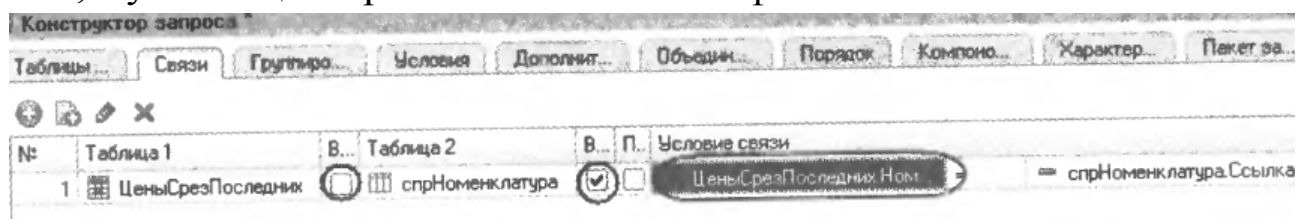
369. В качестве источника данных для запроса выберем объектную таблицу **Номенклатура** и виртуальную таблицу регистра сведений **Цены.СрезПоследних**.

370. Таблицу **Номенклатура** переименуйте в **спрНоменклатура**.

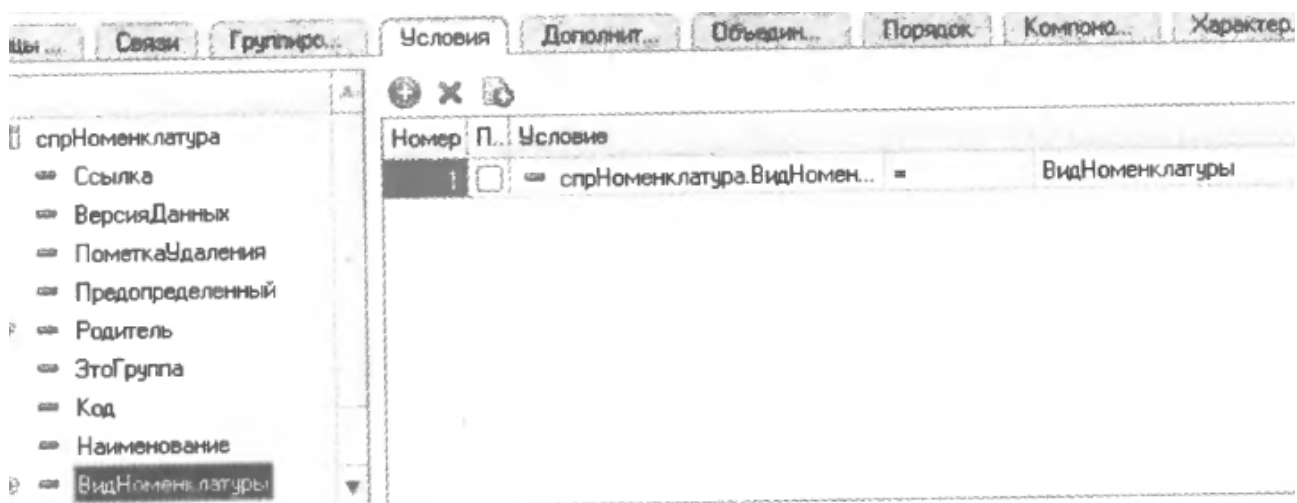
371. В поле **Таблицы** выберите **Цены.СрезПоследних**, нажмите кнопку **Параметры виртуальной таблицы**, в поле **Период** введите **&ДатаОтчета**.

372. Выберите из таблицы поля **спрНоменклатура.Родитель**, **спрНоменклатура.Ссылка**, **Цены.СрезПоследних.Цена**.

373. Выберите вкладку Связи у таблицы регистра снять флажок Все, а у таблицы справочника включить флажок Все.



374. Выберите вкладку условие, выберите поле видНоменклатуры, М2.



375. Выберите вкладку Объединение/Псевдонимы, поле Родитель заменяем на ГруппаУслуг, поле Ссылка на Услуга, нажмите ОК.

ВЫБРАТЬ

спрНоменклатура.Родитель КАК ГруппаУслуг,

спрНоменклатура.Ссылка КАК Услуга,

ЦеныСрезПоследних.Цена

ИЗ

Справочник.Номенклатура КАК спрНоменклатура

ЛЕВОЕ СОЕДИНЕНИЕ РегистрСведений.Цены.СрезПоследних(&ДатаОтчета,) КАК ЦеныСрезПоследних

ПО (ЦеныСрезПоследних.Номенклатура = спрНоменклатура.Ссылка)

ГДЕ

спрНоменклатура.ВидНоменклатуры = &ВидНоменклатуры

376. Выберите вкладку Ресурсы, выберите Цена.

377. Выберите вкладку Параметры, для параметра ВидНоменклатуры в столбце Значение выберите Услуга.

378. Для параметра ДатаОтчета снимите ограничение доступности (Ог), в поле Тип выберите состав даты – Дата.

379. Для параметра Период установите ограничение доступности.

380. Выберите вкладку Настройки, выберите корневой элемент Отчет, МП, Новая группировка по полю ГруппаУслуг, тип группировки Иерархия.

379. Выберите группировку ГруппаУслуг, МП, Новая группировка без указания группового поля (Детальные записи).

380. Выберите Выбранные поля, укажите поля Услуга, Цена.

381. Выберите Другие настройки, общих итогов по вертикали выберите Нет.

382. Выберите вкладку ГруппаУслуг, в списке Расположение полей группировки выберите Отдельно и только в итогах, в поле Заголовок введите Перечень услуг.

383. Выберите вкладку **Параметры**, выберите Дата отчета, нажмите кнопку Свойства элемента пользовательских настроек, включите флажком Включать в пользовательские настройки, нажмите ОК.

384. В окне редактирования объекта конфигурации Отчет Перечень услуг выберите вкладку Подсистемы, выберите Оказание услуг и Бухгалтерия.

385. Откройте периодический регистр Цены, добавьте новое значение для услуги Диагностики на 27 июля – 350 руб.

386. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела Оказание услуг выберите Перечень услуг, введите дату 26 июля, нажмите Сформировать, цена диагностики должна быть 600.

387. Запустим 1С:Предприятие в режиме отладки, в панели действий раздела Оказание услуг выберите Перечень услуг, введите дату 27 июля, нажмите Сформировать, цена диагностики должна быть 350.

11.9. Отчет 5. использование вычисляемого поля в отчете

Отчет Рейтинг клиентов будет показывать в графическом виде, каков доход от оказания услуг каждому из клиентов ООО.

388. Создадим отчет: выберите в дереве объектов конфигурации ветвь **Отчеты**, МП, выберите **Добавить**, в поле **Имя** введите **РейтингКлиентов**, нажмите tab и в поле **Синоним** должно появиться **Рейтинг клиентов**, в поле расширенное представление введите Список оказанных услуг,

389. Нажмите **Открыть** схему компоновки данных. В окне конструктора макета выберите тип макета **Схема компоновки данных**, нажмите **Готово**.

390. Добавьте новый набор данных – запрос: нажмите кнопку **Добавить**, выберите **Добавить набор данных – запрос**.

391. Создайте текст запроса: нажмите кнопку **Конструктор запроса**.

392. В качестве источника данных для запроса выберем виртуальную таблицу регистра накоплений **Продажи.Обороты**.

393. Выберите из таблицы поля

ПродажиОбороты.Клиент

ПродажиОбороты.ВыручкаОборот

ПродажиОбороты.СтоимостьОборот

393. Выберите вкладку **Объединение/Псевдонимы**, поле **ВыручкаОбороты** замените на **выручка**, **стоимостьОборот** на **Стоимость**.

394. Нажмите **ОК**.

395. Выберите вкладку **Вычисляемые поля**, нажмите кнопку **Добавить**, в поле **Путь к данным** введите **Доход**, в поле **Выражение** введите **Выручка - Стоимость**

396. Выберите вкладку **Ресурсы**, выберите **Выручка**, **Доход**, **Стоимость**.

397. Выберите вкладку **Настройки**, выберите **корневой элемент Отчет, МП, Новая диаграмма**.

398. Выберите **Точки, МП, Новая группировка по полю Клиент**.

399. Выберите **Выбранные поля**, выберите **Доход**.

400. Выберите **Другие настройки**, тип диаграммы – **круговая объемная**, в поле **заголовков** введите **Рейтинг клиентов**.

401. В окне редактирования объекта конфигурации **Отчет Перечень услуг** выберите вкладку **Подсистемы**, выберите **Оказание услуг и Бухгалтерия**.

402. Запустим **1С:Предприятие** в режиме отладки, в панели действий раздела **Оказание услуг** выберите **Рейтинг клиентов**, нажмите **Сформировать**.

11.10. Отчет 6. Вывод данных в таблицу

Сделаем универсальный отчет, чтобы позволить пользователю изменять его структуру и внешний вид.

403. Создадим отчет: выберите в дереве объектов конфигурации ветвь **Отчеты, МП**, выберите **Добавить**, в поле **Имя** введите **Универсальный**, нажмите **tab** и в поле **Синоним** должно появиться **Универсальный**, в поле расширенное представление введите **Список оказанных услуг**,

404. Нажмите **Открыть схему компоновки данных**. В окне конструктора макета выберите тип макета **Схема компоновки данных**, нажмите **Готово**.

405. Добавьте новый набор данных – запрос: нажмите кнопку **Добавить**, выберите **Добавить набор данных – запрос**.

406. Создайте текст запроса: нажмите кнопку **Конструктор запроса**.

407. В качестве источника данных для запроса выберем виртуальную таблицу регистра накоплений **Продажи.Обороты**.

408. Выберите из таблицы поля

ПродажиОбороты.Номенклатура

ПродажиОбороты.Клиент

ПродажиОбороты.Мастер

ПродажиОбороты.КоличествоОборот

ПродажиОбороты.ВыручкаОборот

ПродажиОбороты.СтоимостьОборот

409. Нажмите **ОК**.

410. Выберите вкладку **Ресурсы**, выберите **>>**.

411. Выберите вкладку **Настройки**, выберите **корневой элемент Отчет, МП, Новая таблица**.

412. Выберите в структуре элемент **Таблица**, нажмите кнопку **Свойства элемента пользовательских настроек**. Выберите **Выбранные поля, Группировка строк, Группировка колонок**.

413. В окне редактирования объекта конфигурации **Отчет Перечень услуг** выберите вкладку **Подсистемы**, выберите **Оказание услуг**.

414. Запустим **1С:Предприятие** в режиме отладки, в панели действий раздела **Оказание услуг** выберите **Универсальный**, нажмите **Сформировать. Пусто!**

415. В строке **Выбранные поля** нажмите три точки, выберет **выручкаОборот**.

416. В строке Строки нажмите три точки, добавьте группировку по полю Номенклатура с типом Иерархия.

417. В строке Колонки добавьте группировку по полю Мастер.

418. Нажмите Сформировать.

419. В строке Выбранные поля нажмите три точки, выберите еще СтоимостьОборот.

420. В строке Строки нажмите три точки, удалите группировку по полю Номенклатура, введите группировку по полю Клиент.

421. Нажмите Сформировать.

422. В строке Выбранные поля нажмите три точки, удалите СтоимостьОборот.

423. В строке Строки нажмите три точки, удалите старую группировку, введите группировку по полю Номенклатура с типом Только Иерархия.

424. В строке Колонки добавьте группировку по полю Клиент, и поместите ее первой..

425. Нажмите Сформировать.

12-й день. Бухгалтерский учет

12.1. Создание объекта конфигурации План видов характеристик **ВидыСубконто**

Приступим к созданию плана видов характеристик, который будет содержать описания объектов аналитического учета – субконто.

426. Откроем конфигуратор и создадим новый объект конфигурации План видов характеристик. Зададим его имя – **ВидыСубконто**, на вкладке Подсистемы выберите **Бухгалтерия**.

427. Поскольку нам понадобится некий вспомогательный справочник, в котором пользователи будут осуществлять «свободное творчество» по созданию значений новых объектов аналитического учета, создадим объект конфигурации Справочник и назовем его **Субконто**, на закладке **Владельцы** укажем, что этот справочник будет подчинен плану видов характеристик **ВидыСубконто**.

428. В плане видов характеристик зададим тип значения характеристик. Выберите вкладку **Основные** и установим свойство **Тип значения характеристик**, нажмем на кнопку с многоточием и создадим составной тип данных, в который будут входить следующие типы:

СправочникСсылка.Клиенты;

СправочникСсылка.Номенклатура;

СправочникСсылка.Субконто.

Бухгалтерия нашего ООО ведет учет движения денежных средств только в разрезе материалов и клиентов, но не исключено, что в дальнейшем понадобится дополнительная аналитика (поэтому мы используем и справочник **Субконто**). Обратите внимание, что тот справочник, который будет использован в качестве дополнительных значений характеристик, тоже должен входить в составной тип данных типа значений характеристик, иначе конфигуратор выдаст сообщение об ошибке.

Затем укажем, что дополнительные значения характеристик будут находиться в справочнике **Субконто**.

429. На закладке **Прочее**, нажмите **Предпределенные** и начнем ввод предопределенных значений плана видов характеристик.

430. Создадим предопределенный вид субконто: **Материалы**, с кодом 000000001 и типом **СправочникСсылка.Номенклатура**, а затем

создадим вид субконто: **Клиенты**, с кодом 000000002 и типом **СправочникСсылка.Клиенты**.

Создание объекта конфигурации План счетов Основной

Приступим к созданию плана счетов ООО. Бухгалтерский учет в нашем ООО сильно упрощен, и план счетов, по которому работает бухгалтерия, содержит всего 4 счета: **Товары**, **РасчетыСПоставщиками**, **Капитал** и **Дебиторская задолженность**.

431. Откроем конфигуратор и создадим новый объект конфигурации План счетов. Присвоим ему имя – **Основной**, на вкладке Подсистемы укажем **Бухгалтерия**.

432. На закладке **Данные** выделим группу реквизитов **Признаки учета**, нажмите **Добавить**, создадим признак учета **Количественный**.

433. Перейдем на закладку **Субконто** и укажем, что субконто для этого плана счетов будут находиться в плане видов характеристик **ВидыСубконто**. Максимальное количество субконто на счете установим равным **двум**. Также создадим признак учета субконто **Количественный**.

434. Выберите **Прочее**, нажмем кнопку **Предопределенные** и создадим четыре предопределенных счета:

Товары, код **41**, **активный**, с **количественным** учетом в разрезе **материалов**.

РасчетыСПоставщиками, код **60**, **активно/пассивный**.

ДебиторскаяЗадолженность, код **62**, **активно/пассивный**, с учетом в разрезе **клиентов**.

Капитал, с кодом **90**, **активно/пассивный**.

12.2. Создание регистра бухгалтерии Управленческий

435. Откроем конфигуратор и создадим новый объект конфигурации Регистр бухгалтерии. Зададим его имя – **Управленческий**. Укажем, что с ним будет связан план счетов **Основной**. Установим флаг **Корреспонденция**. Он будет говорить о том, что создаваемый нами регистр поддерживает корреспонденции. Это означает, что каждая запись регистра имеет дебетовую и кредитовую часть, что позволит нам получать информацию не только об остатках и оборотах по счетам, но и о корреспонденциях между счетами.

Регистры, не поддерживающие корреспонденцию, применяются тогда, когда не нужно использовать принцип двойной записи, регла-

ментированный в бухгалтерском учете, и, соответственно, контролировать баланс хозяйственных средств и их источников.

436. Теперь перейдем на закладку **Данные** и создадим два ресурса:

Сумма, длина 20, точность 2, **балансовый**;

Количество, длина 15, точность 3, **небалансовый**, признак учета – **количественный**, признак учета субконто – **количественный**.

На этом создание нашего регистра бухгалтерии завершено.

437. Теперь откроем окна редактирования документов **Приходная-Накладная**, выберите вкладку **Движения**, выберите что этот документ будет создавать движения и по регистру бухгалтерии **Управленческий**.

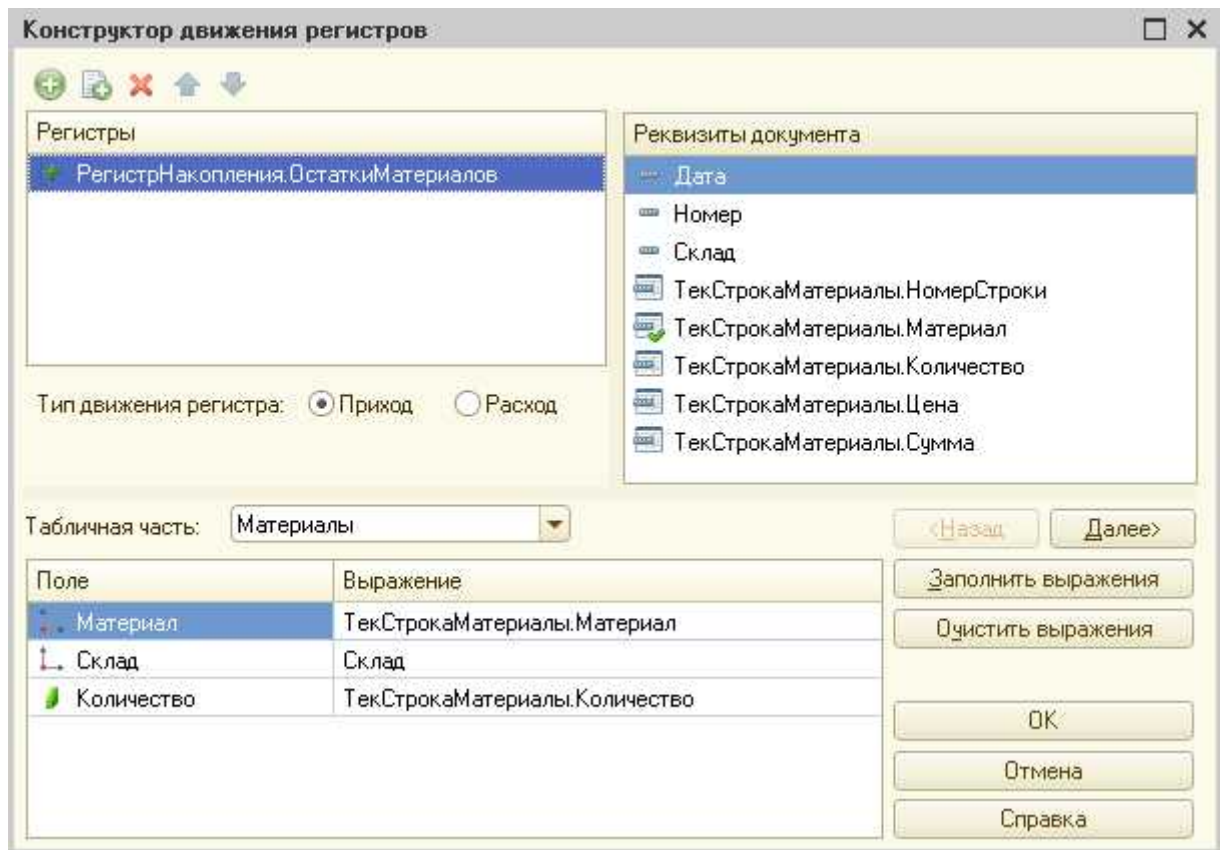
438. Запустим 1С:Предприятие в режиме отладки и откроем регистр бухгалтерии Управленческий. Как видите, платформа (при создании структуры хранения данных) добавила к созданным нами реквизитам регистра еще ряд полей, которые явились следствием использования плана счетов Основной. Прежде всего, это поля СчетДт, СубконтоДт1, СчетКт и СубконтоКт1. Кроме этого, если прокрутить окно вправо до конца, вы обнаружите две колонки Количество: КоличествоДт и КоличествоКт. Для измерений и ресурсов регистра, связанных с признаками учета, платформа создает пару полей для хранения значения ресурса отдельно по дебету и отдельно по кредиту проводки.

12.3. Доработка приходной накладной

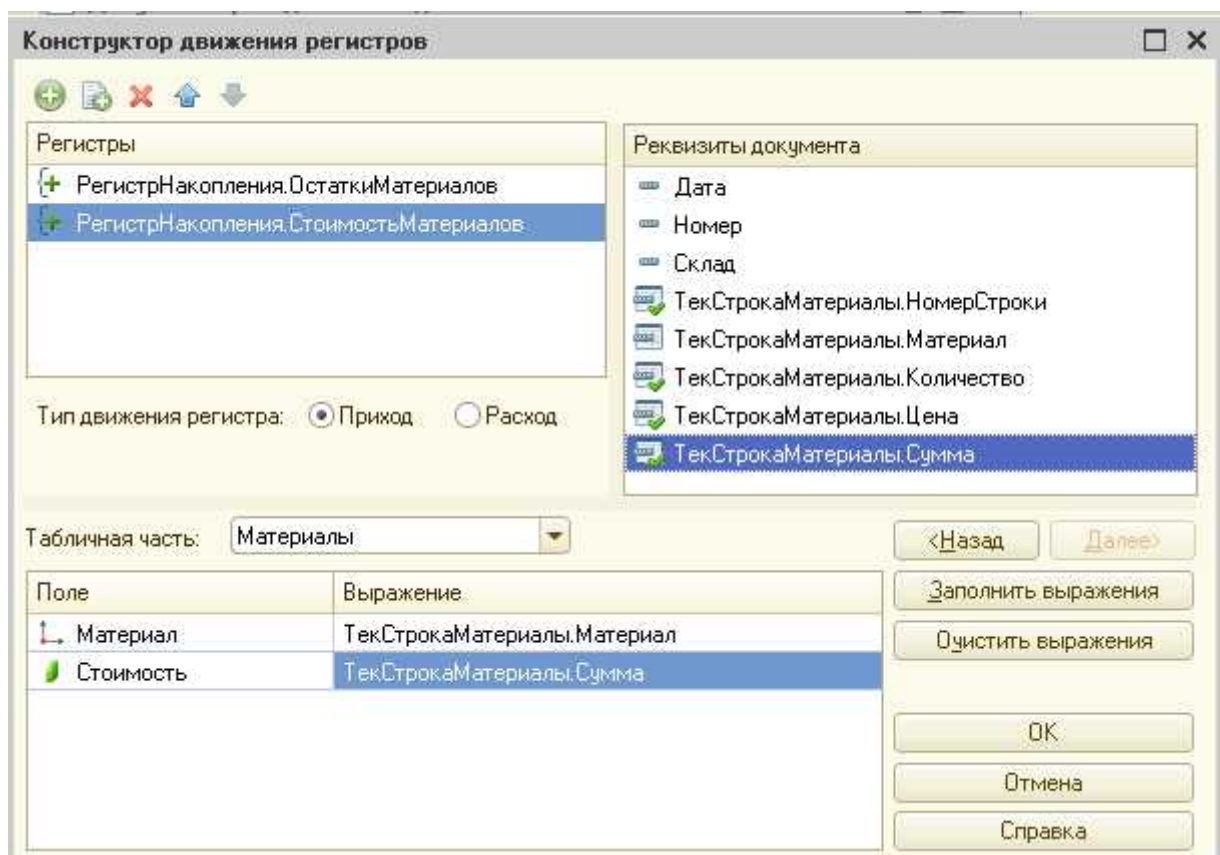
Начнем с простого: доработаем движения документа **Приходная-Накладная**. Для этого нам достаточно будет воспользоваться конструктором движений документа и заменить старые движения документа новыми, по трем регистрам.

439. Откроем конфигуратор. В окне редактирования объекта конфигурации Документ **ПриходнаяНакладная**, на закладке **Движения** запустим **конструктор движений** документа.

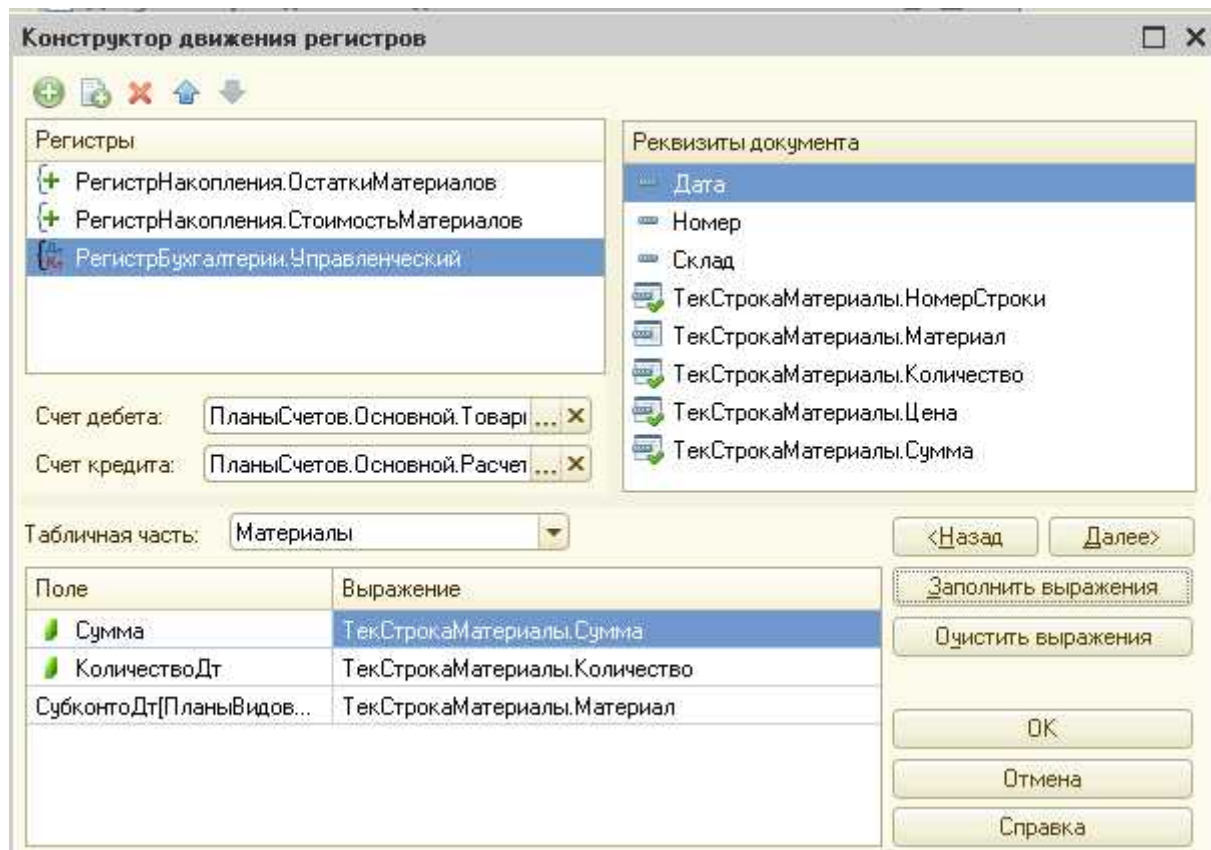
440. Выберите **РегистрНакопления.ОстаткиМатериалов**, заполните по Рис.



441. Выберите **РегистрНакопления.СтоимостьМатериалов**, заполните по Рис.



442. В список регистров добавим РегистрБухгалтерии.Управленческий. В качестве источника данных выберем табличную часть документа ПриходнаяНакладная – Материалы. Счет дебета установим равным ПланыСчетов.Основной.Товары (41), а счет кредита – ПланыСчетов.Основной.РасчетыСПоставщиками (60). Нажмем кнопку Заполнить выражения. См. Рис.



443. Нажмите ОК. В результате процедура Обработка Проведения такая должна быть:

Процедура ОбработкаПроведения(Отказ, Режим)

```
//{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
```

```
// Данный фрагмент построен конструктором.
```

```
// При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
```

```
// регистр ОстаткиМатериалов Приход
```

```
Движения.ОстаткиМатериалов.Записывать = Истина;
```

```
Для Каждого ТекСтрокаМатериалы Из Материалы Цикл
```

```
    Движение = Движения.ОстаткиМатериалов.Добавить();
```


Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
ход;

Движение.Период = Дата;

Движение.Материал = ТекСтрокаМатериалы.Материал;

Движение.Склад = Склад;

Движение.Количество = ТекСтрокаМатериалы.Количество;

КонецЦикла;

// регистр СтоимостьМатериалов Приход

Движения.СтоимостьМатериалов.Записывать = Истина;

Для Каждого ТекСтрокаМатериалы Из Материалы Цикл

Движение = Движения.СтоимостьМатериалов.Добавить();

Движение.ВидДвижения = ВидДвиженияНакопления.Приход;

ход;

Движение.Период = Дата;

Движение.Материал = ТекСтрокаМатериалы.Материал;

Движение.Стоимость = ТекСтрокаМатериалы.Сумма;

КонецЦикла;

// регистр Управленческий

Движения.Управленческий.Записывать = Истина;

Для Каждого ТекСтрокаМатериалы Из Материалы Цикл

Движение = Движения.Управленческий.Добавить();

Движение.СчетДт = ПланыСчетов.Основной.Товары;

Движение.СчетКт = ПланыСчетов.Основной.РасчетыСПоставщиками;

Движение.Период = Дата;

Движение.Сумма = ТекСтрокаМатериалы.Сумма;

Движение.КоличествоДт = ТекСтрокаМатериалы.Количество;
ство;

Движение.СубконтоДт[ПланыВидовХарактеристик.Виды-Субконто.Материалы] = ТекСтрокаМатериалы.Материал;
КонецЦикла;

//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ

КонецПроцедуры

444. Отредактируем командный интерфейс: откройте форму документа Приходная Накладная, в левом верхнем окне выберите Командный интерфейс, в панели Панель навигации раскройте группу Перейти, установите видимость для команды открытия регистра бухгалтерии Управленческий.

445. Запустите 1С:Предприятие в режиме отладки. Откройте документ Приходная накладная, нажмите Провести.

446. Нажмите команду перехода к регистру **Управленческий**, смотрим какое движение сформировал документ в регистре бухгалтерии.

Обратите внимание: поскольку на счете 60 (РасчетыСПоставщиками) отсутствует аналитика и ведется только суммовой учет, в записях движений регистра СубконтоКт1, СубконтоКт2 и КоличествоКт не указаны.

12.4. Оборотно-сальдовая ведомость

Единственный отчет, которым пользуется бухгалтерия нашего ООО, – это отчет Оборотно-сальдовая ведомость.

447. Для того чтобы его сформировать, откроем конфигуратор и создадим новый объект конфигурации Отчет с именем ОборотноСальдоваяВедомость. Создадим новую схему компоновки данных и добавим набор данных - запрос. Откроем конструктор запроса.

Бухгалтерский отчет Оборотно-сальдовая ведомость представляет собой таблицу, в строках которой перечислены все имеющиеся в плане счетов счета, а в колонках – начальное сальдо, оборот и конечное сальдо по дебету и кредиту каждого счета.

448. Поэтому нам для построения такого отчета понадобятся две исходные таблицы: объектная (ссылочная) таблица плана счетов **Основной** и виртуальная таблица регистра бухгалтерии **Управленческий.ОстаткиИОбороты**.

449. Из таблицы Основной мы выберем поля **Код** и **Наименование**, а из таблицы УправленческийОстаткиИОбороты возьмем следующие поля:

СуммаНачальныйОстатокДт;

СуммаНачальныйОстатокКт;

СуммаОборотДт;

СуммаОборотКт;

СуммаКонечныйОстатокДт;

СуммаКонечныйОстатокКт.

450. Перейдем на закладку **Связи** и укажем, что из таблицы **Основной** мы будем выбирать **все** записи, а из **таблицы регистра** – только те, которые соответствуют условию связи, т.е. у **В** снять галочку.

451. Затем на закладке **Объединения/Псевдонимы** зададим псевдонимы полей регистра: **СальдоНачДт**, **СальдоНачКт**, **ОборотДт**, **ОборотКт**, **СальдоКонДт** и **СальдоКонКт**.

452. После этого на закладке **Порядок** укажем, что результат запроса должен быть отсортирован по **возрастанию** поля **Код**. На этом создание отчета закончено, нажмем **ОК**.

Теперь, для того чтобы схема компоновки данных могла отобразить общие итоги по полям бухгалтерских остатков, внесем небольшие изменения в роли, которые она автоматически определила для полей остатка регистра бухгалтерии.

Для этих полей система определила бухгалтерский тип – Дебет и Кредит. Поэтому когда в нашей оборотно-сальдовой ведомости будет рассчитываться общий итог по этим полям, мы получим значение 0, т.к. сумма по дебету счетов будет равна сумме по кредиту, только с обратным знаком. Для того чтобы избежать такой ситуации, в ролях этих полей мы уберем указание бухгалтерского типа и изменим имена групп полей. В этом случае система компоновки данных будет воспринимать эти поля как «обычные» поля остатков.

453. Войдя в режим редактирования поля **Роль** и нажав кнопку **выбора**, можно открыть окно редактирования роли поля.

454. Для полей **СальдоНачДт** и **СальдоКонДт** зададим имя **СальдоДт**, а для полей **СальдоНачКт** и **СальдоКонКт** – **СальдоКт**. Для всех четырех полей установим **Бухгалтерский тип** в значение **Нет**.

455. Перейдем на закладку **Ресурсы** и с помощью кнопки **Добавить** все ресурсы (>>) выберем все доступные ресурсы.

456. На вкладке **Параметры** добавим параметр с именем **Период** и типом **СтандартныйПериод**.

457. Для параметра **НачалоПериода** введите выражение **&Период.ДатаНачала** и запретите его редактирование пользователем.

458. Для параметра **КонецПериода** введите выражение **&Период.ДатаОкончания** и запретите его редактирование пользователем.

459. В заключение перейдем на закладку **настройки** и создадим структуру отчета. Добавим **группировку**, содержащую детальные записи. Затем на закладке **Выбранные поля** выберем все поля для вывода в отчет.

460. На закладке **Другие настройки** укажем заголовок отчета – **Оборотно-сальдовая ведомость**. Для параметра **Расположение общих итогов по вертикали** укажем значение **Начало и конец**.

461. На вкладке **Параметры** выберите для параметра **Период** значение из списка **Этот месяц**.

462. Нажмите кнопку **Свойства элемента пользовательских настроек**, укажите что параметр **Период** будет включен в состав пользовательских настроек, и эта настройка будет находиться в отчетной форме – режим **Быстрый доступ**.

463. Закройте конструктор схемы компоновки данных. Выберите вкладку **Подсистемы**, выберите **Бухгалтерия**.

464. Запустим 1С:Предприятие в режиме отладки и посмотрим, как он работает: выберите **Бухгалтерия**, выберите **Оборотно-сальдовая ведомость**, **Сформировать**.

Приложение 1. Архитектура "1С:Предприятия" как продукт инженерной мысли

Взгляд с другой стороны

За что боремся

Платформа и бизнес-приложения

Метаданные - способ описания бизнес-приложения

Построение приложения на основе модели

Управление данными

Стандартные прототипы прикладных объектов

Прикладные объекты и механизмы

Высокоуровневая модель интерфейса

Интеллектуальные механизмы подготовки отчетов

Построение распределенных и интегрированных информационных систем

Поставка и обновление прикладных решений

А также...

Итак...

Сегодня практически каждый второй продукт человеческой деятельности, предлагаемый покупателю, имеет отношение к высоким технологиям. А это значит, что он не только удовлетворяет конкретные потребности человека, но и является воплощением научной и инженерной мысли. Очевидно, что в некоторых случаях заложенные в продукт идеи и достижения, полностью определяют его сущность и потребительскую ценность.

В этой статье мы решили сконцентрироваться не на описании возможностей продукта "1С:Предприятия", а именно на технологических инновациях, которые в совокупности определили ряд совершенно новых технологий разработки бизнес-приложений и новых качеств самих этих приложений.

Сначала сделаем небольшое замечание по терминологии. Эта статья не ориентирована исключительно на тех, кто имеет дело с системой программ "1С:Предприятие". Как нам кажется, она может быть интересна многим специалистам, интересующимся развитием технологий экономических и корпоративных приложений, независимо от

того с какими продуктами они работают. Поэтому в данной статье мы постараемся лишь в минимальном объеме использовать терминологию, специфическую для "1С:Предприятия". В некоторых случаях мы будем использовать для близких понятий различные термины, приводя их аналоги и необходимые разъяснения.

Небольшое уточнение требуется по самому названию продукта. "1С:Предприятие" это система программ, включающая платформу и набор построенных на ее основе бизнес-приложений (прикладных решений), предназначенных для множества отраслей и предприятий разного масштаба. В данной статье мы будем говорить в основном о платформе, на которой строятся бизнес-приложения:

За что боремся



В начале хочется кратко определиться с тем, что считать основной целью создания таких технологий как платформа "1С:Предприятие". Если посмотреть на историю развития компьютеров и программирования, то можно заметить, что, наряду с повышением производительности, увеличением объемов обрабатываемой информации, повышением эргономичности и т.д. прослеживается достаточно четкое стремление к повышению уровня абстракции программных систем. Эта тенденция является в какой-то степени уникальным свойством, присущим именно компьютерингу, тогда как в других областях человеческой деятельности стратегические цели развития носят более утилитарный характер. Проследить историю повышения уровня абстракции очень легко - начиная от программирования на уровне соединительных шнуров, к машинным кодам, ассемблеру, структурным языкам программирования и т.д. Каждый этап знаменовал повышение уровня абстракции взаимо-

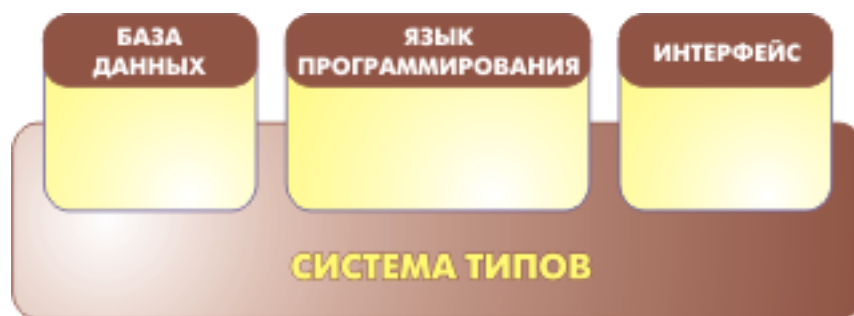
действия человека (и разработчика, и конечного пользователя) с компьютером.

Так вот, основная задача платформы "1С:Предприятие" заключается, прежде всего, в реализации данного подхода при разработке и использовании бизнес-приложений. Разумеется, заодно решаются и традиционные задачи, связанные с производительностью, эргономикой, функциональностью и т.д. Но именно повышение уровня абстракции позволяет перейти от технических и низкоуровневых понятий к более содержательным и высокоуровневым, а значит приблизить их к языку пользователей и специалистов в предметной области. В конечном итоге это позволяет значительно ускорить и унифицировать как саму разработку системы, так и ее сопровождение.

Платформа и бизнес-приложения



Пожалуй, наиболее концептуальными в архитектуре "1С:Предприятия" являются само наличие платформы и понятие бизнес-приложения.



Разумеется, любая фирма-разработчик бизнес-софта имеет достаточно мощный инструментарий и набор технологий, с помощью которых строит свои решения. Однако, при разработке "1С:Предприятия" мы изначально ориентировались на создание полнофункциональной, целостной платформы, которая будет использоваться для построения самых разнообразных бизнес-приложений не только самой фирмой "1С", но и множеством других компаний-разработчиков, знакомых со спецификой тех или иных отраслей. Вот почему платформа изначально создавалась как тиражируемый продукт, включающий как все необходимые технологии для эксплуатации бизнес-приложений, так и инструменты для их разработки и модификации.

В "1С:Предприятии" было введено четкое разделение на платформу и бизнес-приложение. Платформа представляет собой так называемый *framework*, в котором функционирует бизнес-приложение. Мы не смогли найти точного перевода этого слова на русский язык. С одной стороны *framework* можно считать фундаментом для построения приложений, а с другой - средой исполнения. Кроме того, платформа содержит, разумеется, и инструментарий, необходимый для разработки, администрирования и поддержки бизнес-приложений. Такое приложение является самостоятельной сущностью и может выступать в качестве отдельного программного продукта, но полностью опирается на технологии платформы.

Заметим, что понятие платформы и платформенно-ориентированного построения приложений сейчас является общепринятым и трактуется гораздо шире, чем просто способность работать в определенной операционной системе. Под платформой понимается среда исполнения и набор технологий, используемые в качестве основы для построения определенного круга приложений. Фактически, приложения базируются на нескольких платформах, образующих многослойный пирог. При этом важно, что платформа предоставляет разработчику определенную модель, как правило, изолирующую его от понятий и подроб-

ностей более низкоуровневых технологий и платформ. Такова и платформа "1С:Предприятие", позволяющая использовать самые разные технологии более низкого уровня, не меняя кода бизнес-приложений.

Например, она предлагает разработчику собственную модель работы с данными и изолирует его от особенностей конкретного хранилища данных, а это позволяет, не изменяя бизнес-приложение использовать в нем различные хранилища. К примеру, в качестве БД при решении задач небольшого масштаба может применяться собственный файловый движок, а для работы в масштабе предприятия - MS SQL Server.

Ключевым качеством платформы "1С:Предприятие", пожалуй, является достаточность ее средств для решения задач, стоящих перед бизнес-приложениями. Это позволяет обеспечить очень хорошую согласованность всех технологий и инструментов, которыми пользуется разработчик. Ведь часто именно наличие "швов" между различными технологиями становится причиной самых серьезных проблем. Простейший пример - система типов. В платформе "1С:Предприятие" разработчик использует одну систему типов данных и для взаимодействия с БД, и для реализации бизнес-логики, и для построения интерфейсных решений:

Поэтому у него нет проблем, связанных с преобразованием типов при переходах между разными уровнями прикладной системы. Другим очень важным моментом является стандартизация. Наличие единой платформы для большого количества прикладных решений способствует формированию общего "культурного слоя", включающего и людей (программистов, аналитиков, пользователей) и методологию (типовые структуры данных, алгоритмы, пользовательские интерфейсы). Опираясь на этот "культурный слой", разработчик тратит минимум усилий на поиск необходимого решения, практически, в любой ситуации, начиная от включения в проект нового специалиста и кончая реализацией какой-либо подсистемы бизнес-приложения по типовой методологии.

Одним из существенных преимуществ четкого разграничения между платформой и бизнес-приложением является высокий уровень адаптируемости решений под требования клиента.

Следует заметить, что именно для экономических задач особо важна возможность эффективного изменения готового решения разработчиком, не участвовавшим в его создании. В индустрии разработки биз-

нес-приложений, в отличие от многих других областей, существенная часть разработчиков не создает программы "с чистого листа", а дорабатывает и развивает типовые решения.

Это обстоятельство определяет особые требования к обеспечению наглядности и простоте понимания разработчиком существующих решений и максимально учитывается во всех механизмах платформы.

Инструментальные средства "1С:Предприятия" представляют собой не некий дополнительный "toolkit", а являются неотъемлемой составляющей платформы. Они ориентированы в равной степени, как на разработку решений, так и на их адаптацию при внедрении на конкретном предприятии. Эти средства поставляются с каждым комплектом 1С:Предприятия и применяются как для внесения небольших изменений, например, в макет печатной формы, так и для существенной доработки прикладного решения включая структуры данных и бизнес логику. Возможности эффективного внесения изменений в приложение при его внедрении заложены в самих этих инструментах, а, кроме того, этому способствует и архитектура построения прикладного решения. В следующих разделах статьи мы постараемся проиллюстрировать данный тезис.

Выделение бизнес-приложения как самостоятельного элемента, фактически, позволило нам сформировать целую индустрию создания, распространения и поддержки разнообразных прикладных систем, концентрирующую свои усилия на специфике этого класса задач и не требующую глубокого понимания большей части технологических деталей и подробностей. Например, для фирмы, имеющей специалистов, опыт и репутацию в конкретной отрасли, стало вполне реальным в короткие сроки выйти на рынок бизнес-приложений с собственным продуктом, не тратя несколько лет на создание технологической базы.

Метаданные - способ описания бизнес-приложения



В "1С:Предприятии" прикладное решение не пишется в прямом смысле на языке программирования. Язык программирования, конечно, используется, но только там где это действительно необходимо.

В основе бизнес-приложения лежат метаданные. Они представляют собой структурированное декларативное его описание. Метаданные образуют иерархию объектов, из которых формируются все составные части прикладной системы и которые определяют все аспекты ее поведения. Фактически, при работе бизнес-приложения платформа "проигрывает" (интерпретирует) метаданные, обеспечивая всю необходимую функциональность.

Метаданными описываются структуры данных, состав типов, связи между объектами, особенности их поведения и визуального представления, система разграничения прав доступа, пользовательский интерфейс и т.д. В метаданных, фактически, сосредоточены сведения не только о том, "что хранить в базе данных", но и о том, "зачем" хранится та или иная информация, какова ее роль в системе и как связаны между собой информационные массивы. Использование языка программирования ограничено в основном решением тех задач, которые действительно требуют алгоритмического описания, например, расчета налогов, проверки корректности введенных данных и т.д.

Что дает такой подход к построению бизнес-приложения? Во-первых, при описании метаданных широко используется визуальное редактирование. Это позволяет свести существенную часть разработки к визуальному проектированию, не требующему кропотливого написания кода. Однако у данного подхода есть и другие не менее важные преимущества. Описывая прикладное решение в терминах метадан-

ных, разработчик "сообщает" платформе много очень полезной информации, которую та может эффективно использовать в самых различных целях. На основе метаданных система автоматически "выстраивает" большую часть механизмов и объектов, обеспечивающих функционирование прикладного решения. Например, описания метаданных платформе достаточно для того, чтобы автоматически сформировать пользовательский интерфейс системы, обеспечивающий ввод и редактирование взаимосвязанной информации. Другой пример - возможность построения даже конечным пользователем, не имеющим навыков программирования, достаточно сложных отчетов.

Идеология использования метаданных в самых общих словах сводится к простому тезису: "Давайте не будем программировать все функции разрабатываемого решения. Расскажем платформе о составе, структуре, особенностях и взаимосвязи различных его частей, и пусть остальное она сделает сама".

Эта идеология (Metadata Driven) сегодня находит все большее применение во многих перспективных разработках.

Построение приложения на основе модели

В "1С:Предприятии" изначально заложена строгая ориентация на построение прикладного решения на основе определенной модели.

Этот подход является весьма перспективным и по нашей оценке будет доминирующим в обозримом будущем в современных средствах разработки. Идеи построения бизнес-приложений на основе модели, например, нашли воплощение в архитектуре MDA (Model Driven Architecture) консорциума OMG.

Под моделью понимается вся идеология построения прикладного решения. Сюда относятся способы построения структур данных, типы связей между данными, принципы манипулирования данными, формы описания бизнес-логики, способы связи данных с интерфейсными объектами, разделение функциональности по уровням системы и многое другое.

Важно, что все бизнес-приложения неукоснительно следуют принятой модели и этим обеспечивается единообразие и предсказуемость их поведения. Фактически, разработчик, желающий отразить в прикладном решении специфику той или иной предметной области, имеет вполне определенный набор способов решения этой задачи средствами, заложенными в платформу. С одной стороны, такой подход ограничивает (вполне осмысленно) свободу разработчика, но с другой - за-

щищает его от множества ошибок и позволяет в сжатые сроки получать работоспособное решение, которое сможет в дальнейшем развиваться и поддерживаться как им самим, так и, при необходимости, другим специалистом.

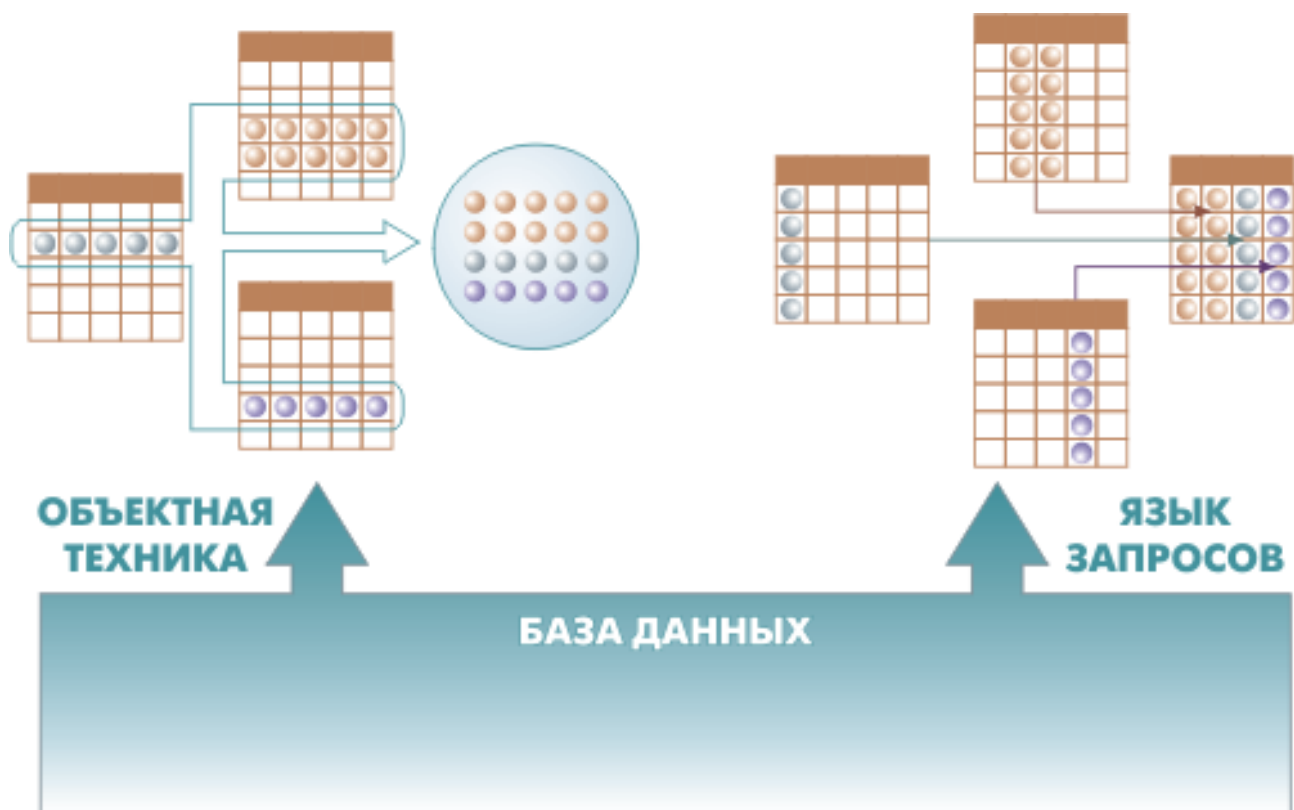
Очевидным следствием этого подхода является изоляция разработчика бизнес-приложения от деталей технологий хранения информации, организации трехуровневой архитектуры и т.д. Например, как уже отмечалось выше, все прикладные решения, базирующиеся на "1С:Предприятии", без каких-либо изменений работают как с собственным файловым движком БД, так и с сервером БД. При этом необходимые структуры данных создаются и изменяются системой автоматически на основе описания метаданных, и разработчику не приходится вникать в детали форматов хранения конкретных СУБД. Манипулирование данными в приложении также описывается в высокоуровневой модели и автоматически исполняется с учетом особенностей используемого хранилища.

По нашей оценке, предоставление разработчику определенной модели, абстрагированной от конкретных используемых средств, будет доминирующим в обозримом будущем в современных средствах разработки.

Наличие единой модели ключевым образом сказывается и на простоте освоения системы. Вся разработка ведется в рамках одной сквозной системы понятий и в едином пространстве типов данных. У разработчика не возникает необходимости осваивать несколько моделей представления и тратить усилия на реализацию переходов между ними на разных уровнях.

Например, описание в метаданных тех или иных объектов (сущностей) сразу определяет и соответствующие типы встроенного языка программирования и необходимые для их хранения структуры БД. Все последующие манипуляции этими объектами, как в памяти, так и в БД выполняются единообразно, не требуя преодоления "барьеров" между различными нотациями, принятыми при работе с СУБД и с универсальными языками программирования.

Управление данными



Одним из ключевых отличий бизнес-приложений от программ иного назначения является парадигма, используемая в них для манипулирования данными.

Очевидно, что для экономических задач работа с данными является основным содержанием с одной стороны и наиболее проблемным вопросом с другой. Это можно объяснить тем, что если во многих областях создания программных систем уже достигнут уровень, при котором программные продукты "близки к совершенству" (например, в текстовых процессорах), то в области экономического софта до этого, мягко говоря, далеко.



Экономические прикладные системы постоянно находятся "под гнетом" противоречивых требований. С одной стороны необходимо обрабатывать большие объемы данных, а с другой - обеспечивать широкую функциональность и высокую производительность. Объемы об-

рабатываемых данных растут, растут требования к разнообразию решаемых задач, и повышаются требования к масштабируемости. Не следует забывать и о повышении удобства разработки, эргономичности систем, возможностях их обновления и доработки и т.д. В современной мировой практике существует несколько парадигм манипулирования данными, которые используются в средствах разработки бизнес-приложений в различных комбинациях. При этом до идеала еще далеко - все имеющиеся и создаваемые технологии являются компромиссными решениями, нацеленными на улучшение показателей систем по нескольким указанным критериям.

Разумеется, эти противоречия касаются всех аспектов системы, но на способах манипулирования данными они проявляются наиболее ярко.

В современной мировой практике существует несколько парадигм, которые используются в средствах разработки бизнес-приложений в различных комбинациях.

В "1С:Предприятии" используется смешанный подход, который с одной стороны имеет много общего с подходами, заложенными в перспективных разработках других фирм, но с другой стороны обладает и существенными отличиями.

Для всех операций модификации данных (создания, изменения и удаления) в "1С:Предприятии" применяется исключительно объектная техника. Это означает, что разработчик взаимодействует с БД не на уровне записей, а с помощью объектов, соответствующих хранимым в БД сущностям. Для изменения хранимых данных, ему не нужно писать сложные запросы и преобразовывать результаты их обработки в объекты языка программирования. Достаточно получить объект из базы данных, изменить его свойства и снова сохранить. Разработчик имеет при этом возможность написать обработчики событий, связанных с изменением данных, выполняя с их помощью различные проверки и изменяя при необходимости другие данные. Система обеспечивает эффективную технологическую поддержку объектного подхода, осуществляя, например, кэширование объектов, контроль объектной и ссылочной целостности и т.д. Для чтения данных может использоваться как объектная техника, так и декларативный язык запросов, который основывается на классическом SQL, но имеет ряд существенных расширений. Расширения направлены с одной стороны на поддержку работы с объектами, хранящимися в базе данных, а с другой -

на эффективное решение экономических задач. Ниже мы рассмотрим язык запросов более подробно:

Одним из ключевых преимуществ объектно-реляционной парадигмы управления данными является наглядность и простота разработки приложений. Объектная техника, используемая в основном для модификации данных, обеспечивает очень хорошую читаемость алгоритмов бизнес-логики, существенно уменьшает количество ошибок при разработке, а также обеспечивает высокий уровень целостности данных. Кроме того, важной особенностью объектной техники является то, что она, благодаря строгой гранулярности манипулирования данными, упрощает переход к распределенным и интегрированным системам.

Декларативное описание запросов, в свою очередь, позволяет получать сколь угодно сложные выборки данных и предоставляет мощные возможности агрегирования, необходимые для решения аналитических задач.

По нашему мнению такая смешанная техника в том или ином виде начнет в обозримом будущем достаточно широко использоваться в перспективных разработках экономических систем.

Здесь, немного забегаая вперед, следует сказать, что в модели "1С:Предприятия" реализована наиболее современная концепция работы с информацией, сочетающая три способа представления данных - хранение сущностей в базе данных, их представление в языке программирования в виде объектов и отображение в формате XML. Фактически любая информация может в зависимости от текущего режима работы представляться одним из этих трех способов.

Долговременное хранение сущностей (persistence) осуществляется в БД, что обеспечивает надежность и эффективную обработку больших объемов информации. Для внесения изменений, данные предварительно преобразуются в объекты встроенного языка. При внутреннем обмене в распределенной БД или взаимодействии с другими информационными системами данные переносятся в формате XML:

Заметим, что все три способа представления опираются на единую систему понятий и от разработчика не требуется усилий для трансформации данных из одного способа представления к другому.

Такой подход позволяет органично вписывать "1С:Предприятие" в будущие гетерогенные решения, которые по нашей оценке (и она совпадает с оценкой многих экспертов) получат со временем самое широкое распространение.

Одной из отличительных особенностей модели "1С:Предприятия", не имеющей, как нам кажется, прямых аналогов в других подобных системах является деление всех прикладных данных на те, что имеют объектную природу и не имеющие таковой. Заметим, что для манипулирования и теми и другими используется объектная техника.

Такое деление, соответствует реальной природе данных. В предметной области всегда есть сущности, имеющие объектную природу, например, "клиенты", "физические лица", "товары". Здесь объект имеет определенную "самодостаточность" не зависящую от данных, которые его описывают. Например, у человека может поменяться фамилия, имя и номер паспорта, но нам важно знать, что это именно то же самое физическое лицо (уникальный объект). С другой стороны есть сущности, не имеющие объектной природы. Например, запись о приходе некоторого товара на некоторый склад является лишь информацией о движении товара и не имеет никакого другого содержания, кроме того, что зафиксировано в записи. Если заменить в такой записи один товар на другой, смысл записи о товародвижении полностью изменится. Иными словами, для такой сущности запись без указания конкретных значений полей не имеет никакого смысла.

В результате такого смыслового разделения сущностей платформа "1С:Предприятие" фактически предлагает разработчику готовую методологию для наиболее естественного решения различных задач манипулирования данными в соответствии с природой этих сущностей.

Работа с объектными сущностями поддерживается представлением сущностей базы данных в виде объектов встроеного языка программирования, а также специальными типами данных, служащими для представления объектных ссылок (ссылок на объекты базы данных). При этом, зная объект, легко получить его ссылку, а, зная ссылку, извлечь из БД объект. Такая техника обеспечивает наглядный и естественный способ описания в исходном коде алгоритмов бизнес-логики, манипулирующих объектами, а, кроме того, гарантирует логическую целостность данных при любых операциях. Она напоминает написание приложений объектных БД, с той лишь разницей, что сохранение данных происходит в таблицах реляционной СУБД. При этом в

модулях, написанных на встроенном языке, может одновременно присутствовать несколько объектов, представляющих одну сущность БД.

Механизмы платформы обеспечивают поддержку уникальных объектных идентификаторов (ссылок), контроль версий объектов, пессимистическую и оптимистическую их блокировку. Оптимистическая блокировка гарантирует логическую целостность изменения объектов, а пессимистическая позволяет организовывать одновременное редактирование пользователями одних и тех же объектов в интерфейсе "1С:Предприятия". Платформа оптимизирует операции считывания объектов за счет использования механизма их кэширования как внутри транзакций, так и вне их. При модификации объектов реализована технология "умной записи": система следит за их изменениями и реально записывает на диск только модифицированные данные, обеспечивая, тем не менее, целостность данной операции.

Работа с необъектными сущностями осуществляется с помощью наборов записей, представляющих собой по сути коллекции записей. Такие наборы поддерживают чтение и модификацию данных с требуемой для прикладной логики гранулярностью.

Важно, что оба указанных подхода позволяют придерживаться простого и естественного стиля написания алгоритмов бизнес-логики. Разработчик, которому необходимо модифицировать имеющуюся в приложении бизнес-логику, читая такой алгоритм, быстро понимает суть дела и может легко производить отладку и корректировку исходного кода. Теперь для взаимодействия с базой данных разработчику не приходится самому программировать сложные преобразования информации из объектных структур в реляционные, что заметно повышает эффективность создания и отладки ПО.

Еще одной важной особенностью объектной техники, принятой в платформе "1С:Предприятие", является то, что те же объекты, которые присутствуют в модулях на встроенном языке (как для объектных, так и для не объектных сущностей) используются и для отображения данных в интерфейсе. Элементы управления форм непосредственно связываются с нужными объектами, и обеспечивают их отображение и редактирование пользователем без какой-либо помощи со стороны разработчика.

Для объектных сущностей платформа "1С:Предприятия" поддерживает механизм представлений. Он отвечает за отображение в интерфейсе значений, заданных ссылками на сущности базы данных. При

необходимости отобразить ссылочное значение система автоматически формирует представление на основании свойств метаданных, оптимизируя, по возможности, получение информации из БД с помощью кэширования и других механизмов. В процессоре обработки запросов и построения отчетов также широко используются представления. Это позволяет универсальным образом получать представления ссылочных полей, если запрос формируется для отображения данных в пользовательском интерфейсе, и автоматически включать отображения представлений в отчеты для полей, содержащих ссылочные значения. Важно, что механизм представлений дает возможность разработчику просто и естественно манипулировать объектными ссылками, минимизируя в то же время число обращений к БД.

Наряду с описанными выше методами манипулирования данными и формирования запросов, система предлагает еще один способ доступа к данным - динамические выборки. Этот механизм позволяет обращаться к очень большим объемам данных, обеспечивая считывание информации порциями. При этом разработчик, только указывает, какие данные необходимо получить, а система автоматически выполняет обращения к БД с необходимой гранулярностью. Важно, что для решения этой задачи не используются какие-либо специфические средства динамического считывания конкретной СУБД, требующие удержания в памяти открытой выборки, а осуществляется автоматическое формирование запросов, последовательно выбирающих блоки записей.

Еще одним важным решением в части работы с данными в "1С:Предприятии" является поддержка в полях таблиц составных типов данных. Эта возможность, насколько нам известно, не имеет близких аналогов в других системах. При описании типа поля какого-либо объекта можно выбрать не только один из доступных типов, но и практически любую (с некоторыми ограничениями) их комбинацию. Например, в поле "Плательщик" в документе, отражающем операцию с банком, допускается хранение ссылки на юридическое или физическое лицо в зависимости от конкретной операции. Хотя приведенный пример является достаточно простым, возможность работы с составными типами позволяет решать такие задачи, как хранение произвольных характеристик товаров, ведение аналитического учета на бухгалтерских счетах по любому составу аналитических разрезов, настраиваемых пользователем и т.д. Важно, что система не просто предоставляет возможность хранения в одном поле разнородных значений, а де-

лает это прозрачным для разработчика способом. Прежде всего, необходимо отметить полную поддержку работы с полями составных типов "движка" базы данных и языка запросов. Для этих полей поддерживается весь набор стандартных операций (сравнение, агрегирование и т.д.). Другим важным моментом является поддержка составных типов в интерфейсных механизмах системы. Например, поле ввода, связанное с данными такого составного типа, предоставляет весь набор возможностей редактирования (выбор типа; редактирование значений всех типов, входящих в указанное поле; ограничение выбираемых типов).

Отдельного внимания заслуживают также средства формирования запросов "1С:Предприятия". Мы уже упомянули, что они основаны на конструкциях стандартного языка SQL, но имеют ряд существенных расширений.

Прежде всего, следует отметить поддержку в языке запросов объектов, хранящихся в БД. Все операторы языка запросов обеспечивают работу со ссылочными типами (полями, хранящими ссылки на объекты БД). Например, поддерживается обращение к полям в нотации "через точку" без ограничения количества уровней. Можно указать в запросе выборку такого поля, как "Товар.Производитель.Страна.Наименование". Для объектов базы данных допускается обращение к вложенным таблицам и как к отдельным таблицам, и как к обычным полям объекта, содержащим коллекции записей.

Другим важным качеством является штатная реализация функции формирования многомерных итогов с произвольным порядком обхода измерений. При этом доступны такие возможности, как сочетание многомерного обхода измерений и многоуровневого обхода иерархии значений каждого измерения (например, многоуровневой структуры подразделений или многоуровневой группировки товаров). Поддерживается также ряд специальных режимов обработки итогов.

Механизм запросов способен на основании свойств прикладных объектов, определенных в метаданных, упорядочивать выборки автоматически. Это позволяет разработчику при создании запроса, предназначенного для получения отчета или визуализации данных, не указывать, по каким полям производить упорядочивание, а, включив автоматический режим, получить стандартную для выбираемых данных сортировку.

Еще одним мощным средством механизма запросов являются виртуальные таблицы. Они обеспечивают доступ к производным данным, предоставляемым различными прикладными подсистемами, не требуя для этого написания сложных запросов. Например, можно обратиться к виртуальной таблице для получения распределения остатков и оборотов товаров по складам и номенклатуре, а также по календарным периодам. Следует отметить, что работа с виртуальными таблицами не является аналогом простого хранения типовых запросов (view). При использовании виртуальных таблиц разработчик указывает набор параметров, описывающих необходимую выборку, беря в их качестве не только конкретные значения, но и, например, сложные условия. Разработчик прикладного решения работает с виртуальной таблицей, практически, так же как и с обычной, но система формирует запрос к БД таким образом, чтобы обеспечить максимальную эффективность. В частности, при обращении к данным, обрабатываемым учетными механизмами, могут использоваться хранимые ими промежуточные итоги.

Стандартные прототипы прикладных объектов



Если говорить о различиях моделей бизнес-приложений и средств их разработки то, пожалуй, наиболее существенным окажется то, в ка-

ких понятиях (можно даже сказать "в какой парадигме") описывается приложение. Разумеется, в каждом средстве разработки может использоваться несколько способов описания, но какой-то один набор понятий всегда является основополагающим.



В качестве примеров существующих подходов можно привести описания в терминах реляционных таблиц, классов объектного языка программирования, сохраняемых сущностей (Entity Persistent) и т.д.

В модели разработки "1С:Предприятия" используется подход, которому мы не нашли явного аналога в других системах. Здесь все прикладное решение описывается метаданными в виде совокупности прикладных объектов, выбираемых из жестко определенного набора прототипов (классов). Можно было бы назвать создаваемые объекты бизнес-компонентами, а их прототипы шаблонами (patterns). Каждый такой прототип отвечает за отражение в прикладном решении определенной совокупности объектов или процессов предметной области, имею-

ших схожие поведенческие характеристики и сходную роль в общей картине решения.

Примерами таких прототипов в "1С:Предприятия" являются "Справочники", "Документы", "Регистры накопления". Каждый прототип имеет некоторую базовую реализацию, которая определяет особенности функционирования, создаваемых на основе данного прототипа объектов: структуру хранимых сущностей вместе с некоторыми предопределенными полями, набор типов языка программирования, методы, свойства и события, а также типовые для решаемой задачи операции, способы отображения и редактирования, методы регулирования прав доступа и т.д.

Если в ходе разработки необходимо создать объект, отражающий особенности предметной области, программист выбирает подходящий прототип и создает на его основе объект метаданных. Далее он может задать различные свойства, определяющие особенности поведения объекта, основанного на этом прототипе, дополнить при необходимости структуру данных объекта, реализовать необходимый набор методов, определить обработчики событий, уточнить способы отображения объекта в интерфейсе и т.д.

Для простых объектов такое доопределение может не потребоваться совсем, и вся процедура создания объекта будет сведена к указанию его имени в свойствах метаданных. В более сложных случаях разработчик имеет возможность достаточно подробно определять и структуры данных, и алгоритмы бизнес-логики, но при этом стандартная функциональность реализуется автоматически без каких-либо усилий с его стороны.

Важно, что количество таких прототипов в платформе не велико, около двух десятков: их легко изучить, с тем, чтобы впоследствии эффективно применять для решения любых задач предметной области.

Таким образом, структура метаданных "1С:Предприятия" является не просто набором описаний объектов в определенных унифицированных терминах. Все прикладное решение, фактически, состоит из объектов, четко разделенных по тем ролям, которые они играют в бизнес-приложении. Такой подход существенно усиливает эффект и от описания системы в терминах метаданных, и от построения приложения на основе модели.

Наличие у объектов, описанных метаданными, стандартной функциональности и предопределенной роли позволяет системе автомати-

чески решать гораздо более широкий круг задач, возлагаемых как непосредственно на эти объекты, так и на общие механизмы, работающие с ними. Фактически, зная о назначении (роли в прикладном решении) того или иного объекта, платформа сама, без какого-либо участия программиста, обеспечивает для него в любой ситуации соответствующий "индивидуальный подход". Например, механизм форм "знает", как лучше редактировать те или иные данные, а стандартный механизм отчетов может автоматически "подстроиться" с тем, чтобы проводить анализ информации наиболее эффективно.

Использование predetermined прототипов обеспечивает стандартизацию модели построения прикладных решений. Такая стандартизация, возможно, не очень существенна для одного отдельно взятого бизнес-приложения, но она имеет огромное значение при создании индустрии разработки и поддержки множества прикладных решений. Стандартизация позволяет существенно упростить задачи проектирования и в несколько раз снижает сложность и трудоемкость сопровождения. Например, если необходимо подключить к проекту еще одного специалиста или передать другому сотруднику поддержку и развитие решения, новый разработчик может после беглого просмотра структуры метаданных буквально за несколько минут составить общее представление о строении и основной функциональности прикладного решения, так как в структуре метаданных все прикладные объекты сгруппированы по принятым в модели "1С:Предприятия" функциональным ролям. В дальнейшем он сможет также быстро разбираться в устройстве и бизнес-логике отдельных подсистем. По опыту эти показатели существенно отличаются от тех, которые достигаются при использовании других подходов к описанию приложения.

Прикладные объекты и механизмы

Выше мы описали только принцип построения прикладного решения, основанный на использовании стандартных прототипов прикладных объектов.

Разумеется, здесь не место для подробного описания всех имеющихся прототипов. Но ниже, чтобы продемонстрировать те идеи, которые заложены в имеющуюся модель построения бизнес-приложения, мы дадим краткий обзор прототипов и предоставляемых ими возможностей.

Сначала мы приведем примеры достаточно простых прототипов - справочников и документов. Справочники описывают каталоги, содержимое которых более или менее постоянно. Это может быть перечень выпускаемой продукции, список клиентов компании, перечень валют и т.д. Справочники обеспечивают поддержку иерархических структур, позволяют относить данные к отдельным объектам и их группам, предоставляют ряд других сервисных возможностей.

Документы отражают в системе события, происходящие в жизни предприятия: поступление материалов, перечисление денег через банк, прием сотрудника на работу и т.д. Этот прототип обеспечивает их отражение в различных учетных механизмах, поддерживает контроль последовательности обработки событий, реализует сквозную нумерацию объектов разного типа и т.д.

Остановимся немного подробнее на некоторых возможностях этих двух прототипов.

Известно, что классическая реляционная модель не имеет готовых средств для поддержки иерархии данных и ее реализация всегда требует кропотливой работы. Справочники (а также некоторые другие прототипы прикладных объектов "1С:Предприятия"), поддерживают многоуровневую иерархию изначально. Включается этот механизм простой активизацией соответствующего свойства в метаданных. При этом поддержка иерархии распространяется сразу на все аспекты использования прикладного объекта. Например, прототип обеспечивает поддержку необходимых свойств и методов в объектной модели манипулирования данными (определение уровня объекта, контроль за цикливанием иерархии и т.д.). В интерфейсных механизмах реализуется представление данных в виде иерархического списка или дерева с навигацией по уровням и интерактивным изменением иерархии. В механизмах отчетов обеспечивается формирование иерархических документов такого рода и получение многоуровневой иерархии итогов в любых отчетах, в которых объекты этого типа, выступают, в качестве параметров для группировки.

Другим примером штатных возможностей прототипов является поддержка механизма проведения документов. Этот механизм предлагает разработчику стандартную модель организации связи между информацией о событиях, происходящих на предприятии, и различными учетными механизмами. Любая вводимая пользователем в виде документов информация может отражаться в любых учетных механизмах

(планировании, управленческом учете, бухгалтерском учете и т.д.). Разработчик должен только указать в свойствах метаданных связь между документами и учетными механизмами, а также описать алгоритм проведения документа. Все необходимые действия по проведению и отмене проведения система будет выполнять автоматически. При этом системой предоставляются дополнительные возможности, такие как поддержка отражения событий в реальном времени, поддержка восстановления последовательности отражения событий, происходящих на предприятии, при изменении их задним числом и т.д. В результате предоставляется единая модель связи исходных данных и учетных механизмов, которая не просто облегчает разработку, но и обеспечивает единообразное предсказуемое поведение всех прикладных решений, что существенно облегчает их освоение и поддержку.

Весьма интересными являются прототипы объектов "1С:Предприятие", применяемые в механизме сложных периодических расчетов. Этот механизм можно рассматривать как универсальный инструмент для решения задач расчета зарплаты любой сложности. На самом деле, расчет зарплаты является наиболее типичным применением данного механизма, но сам механизм не имеет ориентации именно на эту задачу и успешно используется для решения других задач, требующих описания периодических расчетов со сложными взаимосвязями, например, расчета дивидендов, стоимости коммунальных услуг и т.д. Он содержит набор типовых стратегий, примером которых может служить вытеснение одних видов расчетов другими при их пересечении на шкале времени. Применительно к задаче расчета зарплаты эта возможность решает весь комплекс проблем, связанных с пересекающимися по времени начислениями и удержаниями, для которых определены сложные правила взаимного исключения. Другим примером является возможность установки взаимосвязи между различными видами начислений и удержаний по периоду действия. Таким образом, механизм представляет определенную математическую модель, устанавливающую связи между различными видами расчетов и предоставляет разработчику действующие на основе этой модели механизмы обработки информации.

Не углубляясь далее в подробности этого механизма, стоит отметить, что он позволяет полностью описать схему расчета зарплаты на основании предоставляемых прототипов прикладных объектов и заложенных в них возможностей. На примере данного механизма можно

достаточно наглядно продемонстрировать эффект от использования предлагаемого подхода. Мы не имеем на данный момент информации о наличии в каком-либо средстве разработки экономического софта подобного механизма. Во всех известных нам случаях модуль расчета зарплаты (Payroll) реализуется ценой достаточно больших затрат и на проектирование и на реализацию. Причем этот модуль приходится существенно или полностью переписывать для каждой страны, так как он сильно привязан к местному законодательству. Применение механизма сложных периодических расчетов в "1С:Предприятии" позволяет перевести разработку такого модуля от проектирования "с чистого листа" к реализации по стандартной схеме. По нашей оценке это сокращает затраты на порядок, но самое главное, существенно упрощается внесение изменений в процессе поддержки системы.

Еще одним характерным примером служит механизм характеристик, предназначенный для решения задач, которые плохо согласуются с классической реляционной моделью базы данных. Одной из таких задач, является построение структуры для отражения разнообразных свойств (характеристик) номенклатуры товаров или изделий. Если создается решение для одной организации с четкой ориентацией на определенную группу товаров, то все необходимые реквизиты можно заложить непосредственно в структуру каталога товаров. Например, для дилерской фирмы по продаже автомобилей ими будут цвет, объем двигателя и т.д. Однако если создается тиражное бизнес-приложение, или решение для организации, занимающейся производством или продажей разнообразной номенклатуры, этот вариант не подходит, так как очевидно, что состав характеристик должен зависеть от конкретной группы номенклатуры и добавление каждой новой группы не должно сопровождаться доработкой прикладного решения. Очевидно, что реализация такого решения плохо ложится в реляционную структуру и требует особого подхода. Реализованный в "1С:Предприятии" механизм характеристик предоставляет разработчику типовое средство решения подобных задач. Специализированный прототип предназначен для организации хранения списка видов характеристик. При этом каждому такому виду приписывается свой тип данных. Например, объем двигателя будет задаваться числом, а цвет выбираться из справочника. Важно, что конкретные виды характеристик могут не только определяться в самом приложении, но и добавляться пользователем в процессе эксплуатации системы без изменения прикладного решения. При этом разработчик может задействовать данный меха-

низм в самых различных ситуациях, например, для хранения логически связанных комбинаций характеристик номенклатуры и расчета себестоимости в разрезе имеющихся комбинаций.

Еще одним интересным решением является механизм бизнес-процессов (work-flow), позволяющий разработчику организовать совместную работу пользователей при выполнении типовых последовательностей деловых операций. Во многих существующих информационных системах для решения задач work-flow используются специализированные продукты, которые приходится интегрировать с приложениями, решающими экономические задачи. В платформе "1С:Предприятие" механизм бизнес-процессов полностью интегрирован в систему таким образом, что ни разработчик ни пользователь не видят "швов" разделяющих этот механизм и другую функциональность. Этот механизм включает средства для описания в прикладном решении схем бизнес-процессов, и их ролевой маршрутизации, для формирования заданий, выполняющихся в каждой точке маршрута, для управления бизнес-процессом и организации его связи с другими функциями прикладного решения. Данный механизм предоставляет разработчику гибкие возможности управления ветвлением процесса и формирования задач. Например, кроме обычного условного ветвления разработчик может визуально описать параллельное прохождение нескольких веток маршрута, и указать точку их слияния. Допускается направление одного задания группе потенциальных исполнителей, например, если выписать счет должен один из менеджеров отдела. И наоборот, в некоторой точке маршрута можно инициировать несколько заданий, например, если финансовые отчеты должны представить все руководители отделов. Ролевая маршрутизация позволяет формировать задачу не только непосредственно конкретному сотруднику, но и распределять задачи по ролям, подразделениям и другим критериям, которые может описать разработчик прикладного решения. При осуществлении ролевой маршрутизации разрешается указывать текущее распределение обязанностей сотрудников с учетом временных замещений, совмещений нескольких должностей и т.д. Важно отметить, что данный механизм предлагает готовую стратегию автоматизации совместной деятельности работников предприятия. Для описания простейших бизнес-процессов достаточно визуального задания схемы маршрута и указания условий ветвления в их узловых точках. Все остальные действия выполняются системой автоматически. При реализации слож-

ных бизнес-процессов усилия разработчика требуются в основном для тесной их увязки с функциями прикладного решения.

На приведенных примерах мы продемонстрировали, что каждый из прототипов прикладных объектов и объединяющих их механизмов закрывает определенную область задач предметной области существенно снижая затраты на разработку и, что очень важно, обеспечивая стандартизацию прикладных решений.

Чтобы дать представление о спектре инструментов предоставляемых разработчику, ниже мы кратко опишем еще несколько механизмов.

Так, для решения задач, связанных с учетом наличия и движения средств (материальных и денежных), предлагается механизм регистров накопления.

Он поддерживает многомерную систему учета с произвольным составом измерений и ресурсов и обеспечивает оптимизацию получения итогов на различные моменты времени, тем самым позволяя эффективно решать задачи материального учета, производственного и финансового планирования, расчета с контрагентами и т.д.

Механизм бухгалтерского учета представляет собой универсальный "движок" для решения задач автоматизации бухгалтерского учета в различных моделях счетоводства.

Он может использоваться в различных странах и имеет большой запас гибкости для дополнительной настройки. В его основе лежит принцип двойной записи, причем допускается работа с финансовыми транзакциями (операциями), включающими как простые корреспонденции (один дебет и один кредит), принятые в российском учете и в моделях учета некоторых других стран, так и сложные, применяемые в большинстве западных стран (несколько дебетов и несколько кредитов).

Механизм поддерживает многомерную систему учета с произвольным составом измерений и ресурсов. Уникальным его свойством является возможность настройки состава дополнительных измерений независимо для каждого счета. При этом допускается формирование состава измерений как разработчиком при создании прикладного решения, так и пользователем в процессе работы с системой. Многоуровневые и многомерные итоги в разрезе периодов, перекрестных корреспонденций и т.д. генерируются автоматически.

Механизм сведений предназначен для решения задач, связанных с хранением разнообразной информации об объектах в различных разрезах.

Например, данные о ценах на товары могут храниться в разрезе их номенклатуры, категорий клиентов и т.д. При необходимости может поддерживаться хранение истории внесенных изменений, что позволяет получать наряду с актуальными данными, еще и их значения на любой прошедший момент времени. Механизм обеспечивает всю логику работы с хранимыми сведениями, начиная от манипулирования данными до их автоматического контекстного отображения в пользовательском интерфейсе.

Еще раз подчеркнем, что в модели "1С:Предприятия" прикладные механизмы и лежащие в их основе прототипы - это не просто набор шаблонов, которые разработчик может использовать "по мере необходимости". Всё прикладное решение с необходимостью основывается на использовании предлагаемого набора прототипов.

Этого, достаточно компактного, набора вполне хватает, чтобы закрыть все потребности предметной области. Имеющимся набором прототипов система фактически навязывает разработчику стандартную модель проектирования, и это позволяет, по нашей оценке, существенно снизить затраты на построение и поддержку прикладных решений .

Высокоуровневая модель интерфейса

Очевидно, что для бизнес-приложения вопрос организации пользовательского интерфейса имеет особое значение. Во-первых, потому, что для существенной части пользователей эти приложения являются основным инструментом, и такие пользователи работают с ним большую часть дня, не являясь зачастую профессиональными пользователями компьютера. Во-вторых, интерфейс бизнес-приложений обычно очень "объемный". В отличие от многих других классов систем бизнес-приложения, как правило, содержат от нескольких сотен, до нескольких тысяч форм. Кроме того, эти формы могут периодически меняться, например, в ходе развития системы или при необходимости изменения бизнес-логики. Соответственно, затраты на разработку интерфейса прикладного решения весьма высоки. Именно поэтому в платформу "1С:Предприятие" включен целый ряд механизмов, ориентированных на быструю разработку эргономичного пользо-

вательского интерфейса. В них реализована собственная оконная модель, система форм, набор элементов управления и т.д.

Основной идеей построения интерфейса, реализованной в "1С:Предприятии", является максимальное использование информации из метаданных, а также объектов манипулирования данными с тем, чтобы вся конструкция не требовала детальной настройки со стороны разработчика и функционировала по большей части автоматически. Мы уже говорили выше, что объекты встроенного языка, используемые для манипулирования данными, применяются и для их отображения в интерфейсе. Разработчику достаточно связать такой объект с элементом управления и механизм интерфейса полностью возьмет на себя организацию просмотра и модификации данных. Однако, наибольший эффект достигается за счет установления связи между объектом манипулирования данными и самой формой, а не с отдельными элементами управления. В таком случае вся функциональность формы, относящаяся к просмотру и редактированию данных, а также ее связь с другими формами, обеспечивается системой автоматически. Для этого в "1С:Предприятии" реализован механизм расширений форм и расширений элементов управления. Расширения автоматически "включаются в работу" с учетом типа данных, с которыми связан элемент управления или форма.

Например, поле ввода обеспечивает все необходимые действия по редактированию ссылочных значений (выбор из специальных форм, множественный подбор, очистку, подсветку и автоматический выбор незаполненных значений, переход к формам редактирования выбранных объектов и т.д.). Очень эффективен так называемый ввод по строке, который позволяет в несколько раз ускорить массовый ввод информации, исключая необходимость выбора значений из форм списков. Для его реализации в свойствах метаданных задается перечень полей, по которым пользователи могут быстро находить нужные объекты, например, код, артикул, наименование. При наборе в поле ввода нескольких символов система автоматически подставляет то значение, одно из указанных в метаданных полей которого начинается с тех же символов. Если таких значений найдено несколько, пользователю предлагается осуществить выбор из списка. От разработчика при этом не требуется никаких усилий, так как интерфейсный механизм опирается на свойства метаданных.

Другим интересным примером является табличное поле. Этот элемент управления представляет собой мощный механизм визуализации и редактирования данных, предоставляющий пользователю возможность просмотра и редактирования больших списков, за счет динамического считывания. Мы уже говорили выше о механизме динамического считывания, как одном из средств манипулирования данными. В механизме интерфейса "1С:Предприятия" он позволяет просматривать массивы данных практически неограниченного размера без считывания их в память целиком. При этом с точки зрения пользователя навигация по списку практически не отличается от просмотра данных, находящихся в памяти компьютера. Данный механизм поддерживает гибкие возможности поиска и фильтрации, просмотра иерархических структур, редактирование данных в списке и в отдельных формах, настройку состава колонок и их сортировки, печать списков и их экспорт в различные форматы и т.д.

Для форм, списков и других элементов система автоматически формирует командный интерфейс (кнопки, меню и целые панели управления), обеспечивающий все действия по просмотру и редактированию, а также связи с другими формами и сервисные функции.

Описав объект метаданных в прикладном решении, разработчик может вообще не строить для него форму, все необходимые формы будут создаваться автоматически в процессе работы системы. Если же он создает форму, чтобы задать ей специфические свойства, то конструктор генерирует стандартную форму, которая не содержит изначально никакого кода на встроенном языке., так как расширения элементов управления и формы обеспечивают всю необходимую функциональность. Разумеется, разработчик может и без конструктора вставить в любую форму элементы управления, связать их с данными и получить полностью работающую конструкцию, не написав ни строчки кода. Дописывать что-то ему придется лишь тогда, когда он захочет переопределить или расширить стандартную функциональность.

Важно что, интерфейс "1С:Предприятия" обеспечивает работу не просто отдельных форм и элементов управления. Штатная механика управления формами прикладных объектов обеспечивает различные варианты связей между формами используемые при выборе значений, множественном подборе, детализации информации и т.д. Фактически, платформа "1С:Предприятия" предлагает разработчику готовую стратегию организации всего интерфейса бизнес-приложения имеющую

способы реализации практически всех необходимых сценариев работы пользователей.

За счет использования хранящихся в метаданных знаний о связях между различными объектами, поддерживается автоматическая генерация команд перехода между логически связанными формами. Так пользователь, работая с некоторым документом, может без каких-либо усилий со стороны программиста получить доступ к записям, отражающим этот документ в различных учетных механизмах. Кроме связей определяемых в метаданных бизнес-логикой прикладного решения, разработчик может определить (в виде критериев отбора) и дополнительные связи для перехода между объектами, которые так же автоматически будут представляться в командном интерфейсе форм.

Другим стандартным интерфейсным решением является так называемый механизм "ввода на основании". Он позволяет при вводе данных по одному объекту, заполнять автоматически целый ряд полей, извлекая их значения из данных других объектов. Например, выписав счет, пользователь может нажатием одной кнопки сформировать на основании этого счета накладную: для этого достаточно установить в метаданных связь между упомянутыми объектами и описать правила заполнения полей.

Разработчик прикладного решения предоставляется стандартное средство реализации "сменного" командного интерфейса. В прикладных решениях, имеющих большой объем функциональности, это дает пользователю возможность переключаться на другой режим работы с полной или частичной сменой командного интерфейса, не выходя из системы.

Механизм форм обеспечивает стратегию смысловой идентификации форм, позволяющую не открывать повторно уже открытые формы, а активизировать открытые ранее формы при обращении пользователя к той или иной информации.

Таким образом, с помощью расширений форм и других механизмов автоматически поддерживается целостная стратегия пользовательского интерфейса, включающая различные варианты взаимодействия между формами, автоматическое формирование командного интерфейса для навигации по различным режимам и функциям системы, а так же всевозможные сервисные режимы, обеспечивающие комфортную работу пользователя.

Мы уже отмечали, что в интерфейсе "1С:Предприятия" реализовано много специальных элементов, не использующих стандартные средства операционной системы. Это сделано для того, чтобы можно было строить интерфейс, максимально отвечающий специфике деловых приложений.

В частности, в "1С:Предприятии" реализована собственная модель оконной системы, ориентированная на работу с большим количеством разнородной информации. Например, максимизация окна управляется пользователем для каждой формы отдельно, а не как единый режим для всего приложения (как это принято в стандартных MDI приложениях). Кроме традиционных видов окон, в ней используются прикрепленные и прячущиеся окна. В интерфейсе "1С:Предприятия" из модального окна разрешается вызов немодальных окон, благодаря чему, фактически, образуется несколько слоев многооконного интерфейса. Разработчик может открывать модально любые формы, не изменяя логики их работы.

Поддерживается автоматический режим корректировки размеров элементов управления при изменении размера формы и при перемещении разделителей внутри нее. Таким образом, пользователь может установить желаемые размеры для каждой формы и отдельных ее частей, а формы автоматически используют все предоставленное пространство для эффективного отображения информации. Важно, что этот механизм не требует какой-либо настройки со стороны разработчика, но допускает, разумеется, и более тонкий тюнинг.

Формы и элементы управления имеют дизайн, ориентированный с одной стороны на отображение максимального количества информации, а с другой стороны на снижение утомляемости при длительной работе пользователя. Для этого используется "плоский" дизайн интерфейса приближенный к Web-дизайну. Для управления дизайном интерфейса предусмотрен механизм стилей, позволяющий централизованно изменять общий вид прикладного решения. Практически каждый элемент управления "оснащен" функциональностью, ориентированной на специфику экономических задач. Кроме того, предусмотрен ряд элементов управления непосредственно нацеленных на аналитические задачи, например, это диаграмма, диаграмма Ганта и т.д.

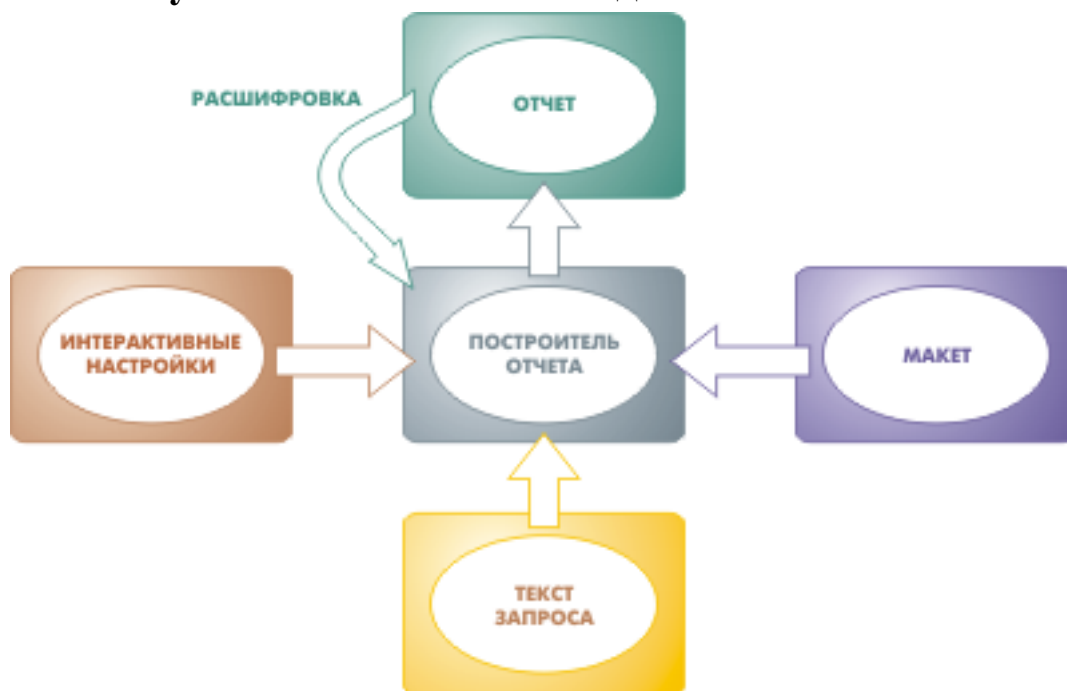
В интерфейсных механизмах "1С:Предприятия" имеется еще много интересных элементов, выше мы перечислили только некоторые решения. Важно, что весь набор интерфейсных механизмов предоставлен

разработчику в достаточно высокоуровневых понятиях. Как уже говорилось, важным качеством концепции разработки в "1С:Предприятии" является изолирование прикладного решения от различного рода технических деталей. Например, разработчик не имеет доступа к управлению движениями мыши или детальной отрисовке элементов управления. Даже достаточно сложные механизмы предоставляются разработчику в таком виде, чтобы с одной стороны они могли бы функционировать вообще без какой-либо настройки, а с другой стороны - обеспечивали бы возможность настройки в простых высокоуровневых категориях, не требующих специальных знаний. Благодаря этому в прикладных решениях поддерживается современный полнофункциональный интерфейс, и в то же время, только незначительная доля исходного кода относится к его визуальной части а большая часть реализует собственно бизнес-логику интерфейса прикладного решения.

Следует упомянуть и о Web-расширении - механизме, позволяющем реализовывать Web-интерфейс к прикладным решениям "1С:Предприятия". Мы не будем рассказывать о деталях технологического устройстве этого механизма. Наиболее интересным в его реализации, на наш взгляд является то, что он также как и основной (rich) интерфейс максимально использует для автоматического формирования Web-форм информацию из метаданных, а также знания о назначении и устройстве прототипов прикладных объектов. Форма для редактирования любого объекта или просмотра списка объектов создается Web-расширением автоматически или может быть определена разработчиком вручную в проекте. Если разработчик описывает форму самостоятельно, то в его распоряжении специализированные элементы управления, которые не только позволяют организовать связь с теми или иными данными, но и предоставляют всю необходимую функциональность по просмотру и редактированию. Например, в полях ввода поддерживается выбор из форм-списков и подбор по первым буквам, в списках поддерживается иерархический просмотр, настройка пользователем фильтрации и сортировки. Система предоставляет большой набор стандартных действий по "обслуживанию" данных и самостоятельно организует связь между формами приложения. Таким образом обеспечивается единообразие двух вариантов пользовательского интерфейса (разумеется, с поправкой на особенности среды Web), а также максимальная автоматизация их разработки, основанная на использовании информации из метаданных.

Одним из наиболее важных, на наш взгляд, следствий построения пользовательского интерфейса на основе стандартных средств предоставляемых платформой является то, что интерфейс всех прикладных решений "1С:Предприятия" построен единообразно. Это позволяет пользователям максимально применять накопленный ранее опыт. Освоив приемы работы с интерфейсом одного бизнес-приложения, пользователь легко начнет работать с любым другим прикладным решением "1С:Предприятия".

Интеллектуальные механизмы подготовки отчетов



В любом бизнес-приложении имеются два основных механизма, обеспечивающих взаимодействие пользователя с системой. Это интерфейс, предназначенный для ввода информации, и средства подготовки отчетов. Поскольку механизмы получения отчетов отвечают, прежде всего, за представление пользователю информации, необходимой для принятия управленческих решений, естественно, что в "1С:Предприятии" им уделено особое внимание.

Технология подготовки отчетов, примененная в "1С:Предприятии", содержит, по нашему мнению, целый ряд уникальных решений, не имеющих прямых аналогов в других продуктах. Прежде всего, следует отметить, что в платформе "1С:Предприятие" нет традиционного для большинства систем отдельного средства "генератор отчетов". Для подготовки отчетов разработчику предлагается целый ряд механизмов, которые могут использоваться в различных комбинациях и в сочетании с другими механизмами. Благодаря этому в "1С:Предприятии" от-

четы органично вписываются в общий интерфейс приложения. Фактически, пользователь в процессе работы не видит грани между общим интерфейсом и механизмом отчетности. По нашему мнению, в современной экономической прикладной системе и не должно быть иначе: ведь аналитическая информация может использоваться во всех режимах и формах, а отчеты формируются по большей части уже не для вывода их на печать, а для интерактивного анализа. Поэтому в "1С:Предприятии" средства подготовки отчетности тесно интегрированы с другими механизмами и имеют мощные возможности для интерактивной работы.

Одним из наиболее интересных механизмов такого рода, предоставляемых платформой, является построитель отчетов, предоставляющий возможность с минимальными усилиями получить отчет с развитой функциональностью.

Нам хотелось добиться того, чтобы разработчик или пользователь кратко описывал, какую информацию необходимо проанализировать, а все остальное для получения полнофункционального управляемого отчета построитель делал бы самостоятельно.

Выборка исходной информации осуществляется на основе описаний, выполненных с помощью языка запросов "1С:Предприятия". Поскольку при этом активно используются готовые виртуальные таблицы учетных механизмов, текст запроса для достаточно сложного отчета будет занимать всего несколько строчек. Его вовсе не обязательно писать вручную. В большинстве случаев запрос формируется средствами визуального конструктора и зачастую вся процедура сводится к выбору источника данных и указанию полей итогов. Данный инструмент позволяет визуально формировать логические конструкции практически неограниченной сложности, включая объединения запросов и вложенные запросы, а также способен "поднимать" для редактирования текст запроса, расположенный в любом месте модуля.

На основании всестороннего анализа текста запроса построитель самостоятельно формирует всю инфраструктуру необходимую не только для генерации отчета, но и для тонкой его настройки пользователем, а также для навигации между отчетами. В результате создается форма, содержащая как собственно отчет, так и разнообразные средства для управления им - выбора включаемых в отчет полей, фильтрации информации по сложным критериям, группировки по строкам и колонкам, настройки упорядочивания данных и т.д. Отображаться от-

чет может в виде таблицы с многоуровневой иерархией строк и колонок, диаграммы, сводной диаграммы или сводной таблицы.

Заметим, что интерактивная настройка фильтрации обеспечивает такие возможности, как установка отбора по полям связанных таблиц (полученных "через точку"), указание в качестве критерия отбора множества значений или групп объектов, в иерархии которых нужно отбирать информацию. Например, пользователь может настроить отчет по закупкам товаров таким образом, чтобы в него вошли только накладные, содержащие товары, производители которых относятся к группам "Отечественные" и "Зарубежные". При этом построитель позволяет сохранять текущие пользовательские настройки отчета для последующей работы.

Кроме того, построитель автоматически включает в сформированный отчет всю информацию, необходимую для расшифровки отдельных ячеек отчета (drill-down). Эта расшифровка выполняется в интерактивном режиме- буквально одним щелчком мыши по выбранной ячейке отчета может быть автоматически сформирован более детальный отчет на основании данных этой ячейки и информации, содержащейся в исходном отчете:

Очень важно, что все эти возможности, представляемые разработчику в готовом виде, могут использоваться им в различных комбинациях. В результате он может гибко определять для каждого создаваемого отчета, какие "степени свободы" предоставить пользователю, указывая это в тексте запроса. Так, например, можно указать, по каким полям пользователь может устанавливать фильтрацию, упорядочивание, группировки и т.д. Сами элементы управления отчетом включаются в форму также по усмотрению разработчика. Поскольку все составляющие данного механизма могут использоваться независимо, разработчик может применять построитель не только для подготовки отчетов, но и для решения многих других задач. Например, он может создать процедуру группового изменения данных, в которой для управления выборкой данных будут задействованы все механизмы построителя, но затем полученная выборка будет не отображена в отчете, а пройдет заданную обработку, например, для выбранной группы товаров будут скорректированы цены.

Так же весьма важным эффектом использования этого механизма мы считаем то, что на его основе в прикладных решениях "1С:Предприятия" реализуются инструменты, в которых "продвинутые" пользо-

ватели действительно могут самостоятельно создавать достаточно сложные отчеты. На основе этого механизма построена специальная консоль отчетов, используемая в большинстве прикладных решений и предоставляющая пользователю возможность проектировать собственные отчеты, тут же их формировать, настраивать и даже создавать систему связанных, детализирующих друг друга отчетов.

Не были забыты и вопросы эффективности изобразительных средств. В "1С:Предприятии" для визуализации и печати отчетов используется прежде всего метафора табличного документа (без средств вычислений). Основной идеей данного решения является использование модели электронной таблицы (spreadsheet) в качестве средства оформления отчетных документов, без использования ее как средства вычисления.

В отличие от большинства электронных таблиц табличный документ "1С:Предприятия" может содержать таблицы, имеющие очень большие размеры, как по вертикали, так и по горизонтали. У него богатый набор изобразительных возможностей, позволяющий с одной стороны создавать хорошо оформленные аналитические отчеты, а с другой - подготавливать регламентированные бланки с точной привязкой содержащихся в них объектов к локальным координатам. В отличие от стандартных электронных таблиц здесь разрешается описывать строки с различными настройками ширины колонок. Это необходимо для создания сложных отчетов включающих несколько разноформатных таблиц.

В большинстве случаев заполнение отчета выполняется на основании макета. Для этого в табличном документе предусмотрена гибкая система именования областей и описания параметров. Разработчик визуально проектирует макет, определяющий внешний вида отчета, выделяя в нем различные области, которые должны заполняться данными, и описывая их параметры. При формировании отчета происходит совмещение областей с данными и таким образом содержательная часть отчета объединяется с его внешним оформлением. Следует заметить, что макет отчета может как формироваться строителем автоматически, так и создаваться разработчиком. Допускается использование готовых стилей оформления отчетов и проектирование собственных.

Важно, что средство генерации табличных документов служит не только для получения статических печатных форм. В нем имеется

мощный интерактивный механизм для управления многоуровневыми группировками, расшифровки ячеек отчета(drill-down), автонастройки ширины колонок и т.д. Допускается сохранение отчета в различных форматах (html, xls, txt). Кроме того, для интерактивного анализа многомерной информации в табличном документе может использоваться сводная таблица, которая позволяет непосредственно управлять группировкой информации и составом отображаемых данных, не обновляя отчета.

Чрезвычайно важным качеством табличного документа является возможность использования его в качестве полнофункционального механизма ввода и редактирования информации. Такой документ может содержать любые элементы управления, используемые в формах "1С:Предприятия", как в самих ячейках и рисунках, так и поверх ячеек. Это особенно полезно в тех случаях, когда одна и та же форма должна представлять собой и отчет, и инструмент для редактирования информации. Примером может служить отчет о плановых показателях, которые разрешается корректировать.

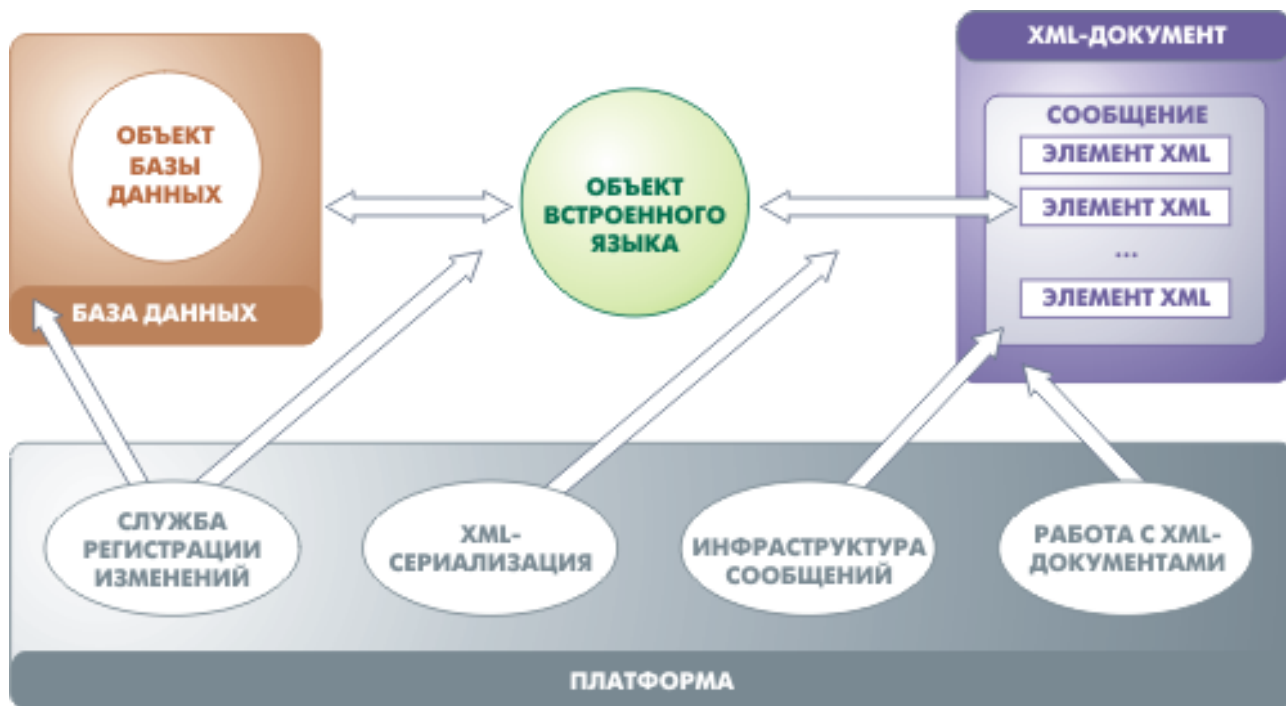
Для визуального представления аналитической информации разработчику предлагается набор различного вида диаграмм, а также диаграмма Ганта, сводная диаграмма и дендрограмма. Причем эти элементы могут включаться как в формы, наравне с другими элементами управления, так и в табличные документы для получения комплексных аналитических отчетов.

Весьма полезен также механизм интеллектуального анализа данных (data mining), с помощью которого прикладное решение можно с минимальными затратами оснастить такими аналитическими инструментами, как кластерный анализ, дерево решений и т.д.

Важная особенность платформы "1С:Предприятие" состоит в том, что для формирования отчетов любой сложности разработчику не придется подключать дополнительные внешние механизмы. При этом все внутренние инструменты хорошо интегрированы, базируются на единой системе понятий и максимально используют возможности друг друга. Например, сводная таблица, располагаемая в табличном документе, для того чтобы автоматически модифицировать запрос к базе данных по мере того, как пользователь визуально указывает новые уровни группировки, обращается к функциям построителя отчетов. Разумеется, в системе присутствует и конструктор отчетов, который позволяет проектировать их интерактивно, однако важно, что конструктор

тор только "собирает" отчет из имеющихся механизмов ("кубиков"), применяемых и для решения множества других задач. Как показала практика, при таком подходе на разработку достаточно сложного отчета уходит не более нескольких минут.

Построение распределенных и интегрированных информационных систем



Еще одной "козырной картой" платформы "1С:Предприятие" являются механизмы обмена данными.

Изначально решение о создании механизма обмена данными, предназначенного для территориально распределенных информационных баз и не требующего постоянного соединения, было обусловлено в большей степени отсутствием в распоряжении отечественных предприятий качественных высокоскоростных телекоммуникационных каналов. Однако некоторое время спустя стало очевидно, что разработка таких механизмов лежит в русле одного из наиболее перспективных направлений мировой инженерной мысли.

Сегодня в распоряжении разработчика приложений имеется мощный набор механизмов обмена, способный решать самые разнообразные задачи. Кроме традиционной поддержки территориально распределенных информационных систем, с его помощью осуществляется интеграция с другими прикладными программами, а также строятся сложные гетерогенные информационные системы, включающие наря-

ду с решениями на платформе "1С:Предприятие" еще и внешние приложения.

Использование технологий обмена для решения столь широкого спектра задач объясняется тем, что в распоряжении разработчика не монолитное решение, а набор механизмов, которые могут применяться как по отдельности, так и совместно, в любых сочетаниях. Эти механизмы поддерживают работу с XML документами, XML-сериализацию данных, инфраструктуру сообщений, службу регистрации изменений, планы обмена, управление распределенной информационной базой:

Практически уникальным, по нашему мнению, качеством данного набора механизмов является то, что он обеспечивает очень высокий уровень готовности системы к работе в распределенной среде - организация обмена практически не требует дополнительных затрат на разработку. Нужно просто задать в интерактивном режиме состав данных, участвующих в обмене, а механизм обеспечит формирование сообщений и их загрузку. При этом система автоматически организует обмен только измененной информацией, отслеживает получение сообщений, определяет необходимость повторной отправки данных, разрешает коллизии и проверяет целостность загружаемой информации. Гибкие возможности настройки позволяют сформировать практически любую топологию схемы узлов обмена (звезда, снежинка, схемы без центрального узла). Состав данных, участвующих в обмене, и правила разрешения коллизий могут задаваться произвольно. При этом механизмы обмена с одной стороны минимизируют объем передаваемых данных (пересылаются только измененные данные), а с другой - гарантируют устойчивость к потере сообщений. Иными словами, система способна функционировать как в условиях гарантированной доставки сообщений, так и без таковой.

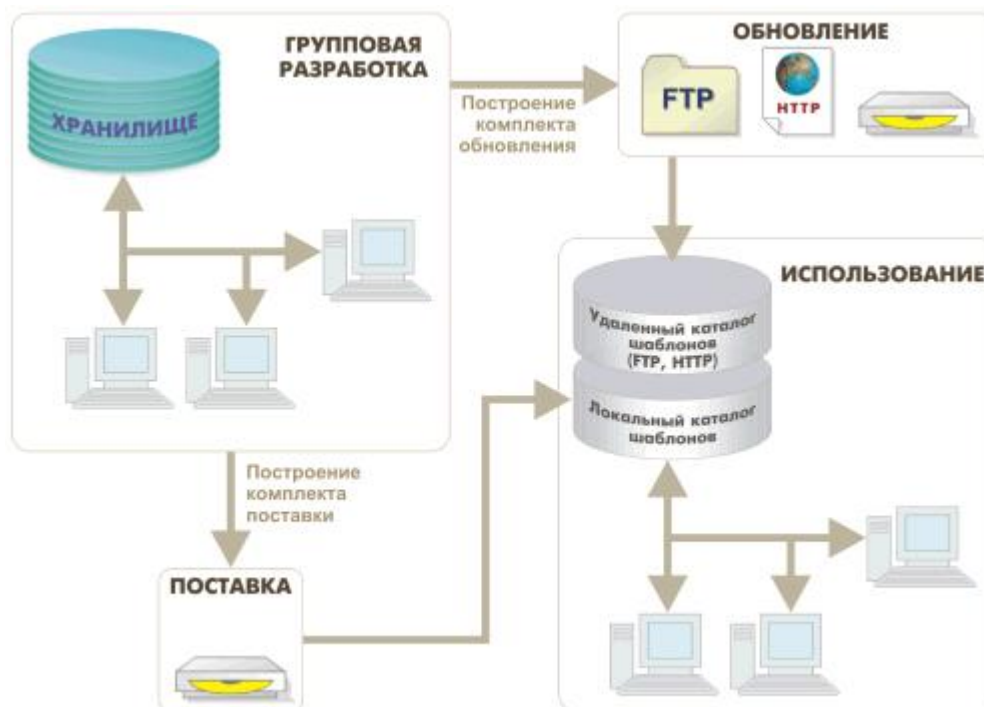
Наличие в платформе эффективных, не требующих сложной настройки механизмов обмена обязано, прежде всего, общей архитектуре построения прикладных решений, реализованной в платформе "1С:Предприятие". Мы уже говорили, о том, что органичный переход к распределенным и интегрированным системам обеспечивается объектной техникой манипулирования данными, используемой в "1С:Предприятии". Система соответствующих объектов обеспечивает штатные возможности сохранения (persistence) любых прикладных данных в формате XML. Кроме того, благодаря наличию стандартных прототипов прикладных объектов платформа способна автоматически

задавать для каждого объекта необходимую гранулярность передачи изменений и стратегию обмена в соответствии с его назначением. Основной проблемой большинства известных систем обмена и репликации является то, что в заложенной в них семантике описания данных не содержится сведений о том, какими порциями выполнять обмен, как разрешать коллизии, как обеспечивать логическую целостность и непротиворечивость данных. Из-за этого разработчику при использовании подобных механизмов обычно приходится детально описывать процедуры обмена для каждого информационного массива.

Механизмы обмена "1С:Предприятия" функционируют на уровне стандартных прототипов прикладных объектов и поэтому имеют в своем распоряжении всю необходимую информацию такого рода изначально. Разработчику необходимо только указать, что в обмене участвуют объекты определенного вида. Далее все действия, по регистрации изменений, формированию сообщений, загрузке данных и разрешению коллизий будут выполняться автоматически. Разработчик, конечно, имеет возможность вручную выполнить "тонкую настройку", но она потребуется только в отдельных случаях.

Другая важная особенность механизмов обмена "1С:Предприятия" - их соответствие передовым мировым концепциям интеграции информационных систем. Мы прекрасно понимаем, что сегодня ни один разработчик экономического софта не может считать себя "царем горы", которому нет необходимости заботиться о тесном взаимодействии с системами других вендоров. В данном контексте взаимодействие - не просто умение вызвать какую-то функцию из другой программы или экспортировать в нее какие-либо данные. Речь идет о построении сложных гетерогенных систем, в которых несколько разнородных прикладных решений образуют слаженно действующий ансамбль. Очевидно, что эта тенденция станет в обозримом будущем одной из доминирующих на рынке экономических систем. Механизмы обмена "1С:Предприятия" имеют высокую готовность к работе в гетерогенных системах сегодняшнего и завтрашнего дня не только за счет того, что весь обмен производится на формате XML, но, что еще более важно, в силу идеологического соответствия реализованных решений указанной тенденции. Уже сегодня механизмы обмена "1С:Предприятия" позволяют осуществлять взаимодействие не только с отдельными приложениями, но и с перспективными интеграционными платформами.

Поставка и обновление прикладных решений



Практически все производители прикладных программ предусматривают в них механизмы для обновления версий. Однако для таких систем как "1С:Предприятие" подобные задачи представляют особую сложность. Это объясняется, во-первых, тем, что экономические системы обновляются гораздо чаще, чем другие, а во-вторых, тем, что достаточно часто практикуется изменение прикладных решений "1С:Предприятия" "на местах" с целью адаптации тиражного продукта к потребностям конкретного клиента. Последняя особенность представляет особую сложность, так как при выпуске разработчиком любого обновления прикладного решения возникает необходимость синхронизации сделанных им изменений с теми, что были внесены в процессе внедрения у заказчика.

Для решения этих и множества других задач, связанных с поставкой и обновлением прикладных решений, в "1С:Предприятии" реализован целый комплекс механизмов ориентированных как на разработчиков, так и на пользователей. Они охватывают весь технологический процесс поставки и поддержки: подготовку дистрибутивных файлов, инкрементных обновлений и установочных комплектов поставки, публикацию обновлений в Интернете, автоматический их поиск и выполнение, управление составом поддержки на уровне объектов конфигурации и т.д.:

Одним из наиболее существенных элементов технологии обновления является механизм, обеспечивающий синхронизацию изменений, сделанных поставщиком прикладного решения с изменениями, внесенными при внедрении на конкретном предприятии. Он предоставляет мощные функции сравнения и анализа изменений, а также средства управления их синхронизацией. Администратор или разработчик может детально настроить синхронизацию обновлений вплоть до отдельных объектов, отдельных свойств и отдельных процедур модулей. Например, если специалист, отвечающий за сопровождение прикладного решения на предприятии, отметит объекты, которые намерен поддерживать самостоятельно, они не будут в дальнейшем обновляться при установке очередного обновления от поставщика. Если объекты необходимо объединить, то для упрощения синхронизации изменений можно настроить приоритеты такого объединения.

Механизмы обновления позволяют построить сложную многоуровневую систему сопровождения. Поставщики специализированных вертикальных решений могут находиться на поддержке у разработчиков универсальных приложений, и обеспечивать в свою очередь поддержку собственных клиентов. Пользователь при этом будет эксплуатировать прикладное решение, отдельные части которого сопровождаются разными фирмами.

Поскольку механизм обновления в значительной мере основан на том, что все прикладное решение описывается с помощью метаданных, а бизнес-логика приложения строится на стандартных прототипах прикладных объектов, система способна поддерживать стратегию обновления с учетом специфики архитектуры прикладного решения, контролировать его логическую целостность и предоставлять пользователю информацию об изменениях в наглядной форме.

А также...

Допускается реализация всего интерфейса прикладного решения на нескольких языках. Каждый пользователь может работать с одним из заложенных в него языков. При этом не нужно выносить все строки в отдельные файлы. Элементы интерфейса редактируются на нескольких языках "по месту" - в формах, макетах печатных документов, в

меню и т.д. Разработчик может в любой момент переключиться и редактировать интерфейс форм на другом языке. Допускается также очередное редактирование конкретной надписи на всех поддерживаемых языках. Если необходимо расширить языковую базу, можно собрать все текстовые элементы в общий список с тем, чтобы перевод одинаковых надписей отражался сразу во всех компонентах интерфейса. Общая стратегия интернационализации включает также перевод системного интерфейса, хранение всех строк в кодировке UNICODE, форматирование дат и чисел в соответствии с особенностями различных стран и языков, построение правил сортировки с учетом национальных стандартов и т.д.

Выполнение алгоритмов бизнес-логики разработчик может по своему усмотрению переносить на сервер "1С:Предприятия". Это позволяет ему управлять распределением нагрузки между клиентом и сервером. При этом от разработчика не требуется специальных навыков построения трехуровневых архитектур, знания сетевых протоколов и т.д. Все технологические детали система берет на себя и обеспечивает рациональное использование серверных ресурсов за счет поддержки stateless-модели, кэширования и разделения системных ресурсов и т.д.

В системе защиты прав доступа "1С:Предприятия" реализована возможность настройки ограничений на уровне отдельных записей, а также полей таблиц. Это позволяет, например, предоставить пользователю доступ к именам сотрудников только в пределах своего департамента, а к данным по окладу и семейному положению - только для своих непосредственных подчиненных. Настройка таких ограничений требует лишь задания формального условия, после чего система контролирует все действия пользователей автоматически, разработчику не нужно учитывать эти ограничения в явном виде при реализации различных интерфейсных режимов и алгоритмов бизнес-логики.

Обеспечиваются две стратегии проверки прав доступа. В зависимости от решаемой задачи, система контроля прав доступа может как отвергнуть обращение пользователя к закрытым для него данным, так и просто исключить недоступные записи из выборки, предоставив пользователю только доступную информацию.

Одним из мощных инструментов разработчика в "1С:Предприятии" является механизм сравнения и объединения прикладных решений. Мы уже упоминали о нем, когда говорили о возможностях поддержки, однако этот механизм может весьма эффективно использоваться и в процессе разработки. Данный инструмент может использоваться как для анализа различий, так и для переноса части функций из одного приложения в другое. Он обеспечивает удобное визуальное представление различий между прикладными решениями и имеет гибкие возможности настройки логики сравнения. При сравнении родственных конфигураций данный механизм автоматически сопоставляет даже переименованные объекты, используя для этого внутреннюю идентификацию метаданных. Соответствие сравниваемых объектов независимо от совпадения их имен можно настроить и вручную, после чего оно будет учтено и во всех ссылках на такие объекты, имеющихся в других объектах прикладного решения. Кроме того, механизм сравнения содержит средства для визуального представления отличий форм интерфейса и макетов печатных документов. Благодаря использованию структуры метаданных и средств визуального сравнения интерфейсных объектов, с помощью нашей платформы удастся решать гораздо больше задач, чем в тех системах, где механизмы сравнения базируются на сопоставлении файлов исходных кодов программ.

В "1С:Предприятии" поддерживается протоколирование действий пользователей в журнале регистрации. Система имеет несколько уровней протоколирования, которые могут функционировать автоматически, не требуя дополнительных усилий со стороны разработчика. Прежде всего, разумеется, фиксируются начало и окончание сеансов работы пользователей, административные операции с информационной базой, а также регистрируемые ошибки. За счет того, что все изменения в БД выполняются исключительно в объектной технике (через объекты, отвечающие за манипулирование данными), система может автоматически регистрировать их в журнале, причем будет делать это независимо от того, как они выполнялись (интерактивно или программно). Разработчик может также реализовать внесение в журнал любых других записей, которые система не способна фиксировать автоматически, например, информацию об отправке факса или формировании отчета. Важно заметить, что журнал регистрации реализован как отдельный системный механизм. Он не использует для хранения ин-

формации базу данных "1С:Предприятия", а, следовательно, его ведение не создает существенной дополнительной нагрузки на систему и не замедляет работу пользователей.

Итак...

В этой статье мы попытались перечислить наиболее интересные технологические новшества платформы "1С:Предприятие". При этом мы старались показать, что все реализованные в ней решения основаны на использовании единой модели и единой архитектуры.

Важно и то, что все механизмы платформы в той или иной степени используют возможности других механизмов. Например, почти все они опираются на структуру метаданных и наличие системы прототипов прикладных объектов. Таким образом, хотя каждая из возможностей "1С:Предприятия" может быть достаточно ценна сама по себе, наибольший интерес представляет рассмотрение всей их совокупности в рамках целостной модели.

Сформулируем еще раз несколько общих принципов, лежащих в основе модели "1С:Предприятия":

- * Архитектура платформы и ее инструментальные средства реализованы таким образом, чтобы разработчик бизнес-приложения мог максимально абстрагируясь от низкоуровневых технологий, сосредоточиться на решении прикладных задач своей предметной области.

- * Вся разработка от построения структур данных, до проектирования элементов интерфейса и подключения средств интеграции ведется в одной системе понятий, что существенно ускоряет обучение специалистов и повышает производительность их труда.

- * Платформа содержит готовые ответы практически на все вопросы, возникающие у разработчика прикладного решения, начиная от того, как отражать в информационной базе данные предметной области и кончая процедурами поставки, поддержки и администрирования. Соответственно, разработчику не требуется осваивать разнородные технологии и решать вопросы их совместного использования.

Наличие единой, сквозной высокоуровневой модели и реализующих ее технологий позволяет, по нашей оценке, на порядок сократить затраты на создание и поддержку прикладных решений.

Начиная в 1996 г. работы в этом направлении, и даже выпустив первую версию платформы, мы, конечно, не были абсолютно уверены в правильности выбранного пути. За прошедшее с тех пор время концепция системы успешно развивалась, и сегодня она воплощена в платформе "1С:Предприятие 8". Строго говоря, в мире не так уж много специализированных технологий, ориентированных именно на быстрое создание бизнес-приложений. Однако становится все более очевидным, что данный путь лежит на стратегическом направлении развития бизнес-софта.

По оценкам многих специалистов, с которыми нам приходилось общаться, ситуация в России и в странах бывшего СССР, где сложилась целая индустрия разработки и поддержки бизнес-решений, основанная на специализированной технологической платформе, по мировым меркам является уникальной. По нашим наблюдениям, не существует аналогичной специализированной платформы, на которой различными фирмами создавалось бы такое количество разнообразных массовых бизнес-приложений, покрывающих потребности существенной части рынка какого-либо региона. Соответственно, для многих специалистов наши подходы и решения представляют особый интерес, так как реальный эффект от их реализации хорошо заметен на практике.

Разумеется, все изложенные выше соображения (включая и анализ современных тенденций) являются только нашей позицией и результатом наших исследований. Мы ни в коем случае не претендуем на абсолютную истину, которой в этих вопросах, по-видимому, и не существует. Наверняка, у многих читателей есть своя, отличная от нашей, точка зрения на проблемы, затронутые в статье. Мы будем признательны за отзывы (в том числе критические) по данной статье.

Сергей Нуралиев,

руководитель отделения разработки экономических программ фирмы «1С».

Литература

1. Кравченко Т.К., Пресняков В.Ф. Информационные технологии управления предприятием. — М.: ГУ-ВШЭ, 2002.
2. «1С» подвела итоги работы в 2006 году. <http://business.compulenta.ru/300249/>
3. Орлов П. Утер нос Биллу Гейтсу. Оказывается, у нас есть не только нефть и газ, но еще и мозги. Газета Труд № 057 за 05.04.2007.
<http://www.trud.ru/trud.php?id=200704050570601>
4. Конфигурация «Управление торговлей». <http://v8.1c.ru/trade/>
5. Установка 1С:Предприятия v8.0.
http://www.script-coding.info/v8/v8_Setup.html
6. Шуремов Е. Л. Информационные технологии оптовой торговли: Практическое пособие. — М.: ООО "1С-Публишинг", 2003. —186 с.
7. Шустикова Т.Б. 1С:Предприятие 8.0. Управление торговлей. — М: НТ Пресс, 2005. — 284 с.
8. Радченко М.Г. 1С:Предприятие 8.2ю Практическое пособие разработчика. Примеры и типовые приемы. – М.: ООО «1С-Публишинг», 2009. – 874 с.
9. Нуралиев. С. Архитектура "1С:Предприятия" как продукт инженерной мысли. <http://v8.1c.ru/metod/architecture/>

Оглавление

Введение.....	3
1-й день. Начало.....	4
1.1. Установка платформы.....	4
1.2. Программирование или разработка.....	8
1.3. Общие сведения о системе 1С:Предприятие.....	9
1.4. Конфигурация и прикладное решение.....	10
1.5. Режимы работы системы.....	12
1.6. Создание новой ИБ.....	13
1.7. Дерево объектов конфигурации.....	13
1.8. Объекты конфигурации.....	14
1.9. Что такое подсистема.....	17
1.10. Добавление подсистемы.....	18
Контрольные вопросы.....	19
2-й день. Справочники.....	21
2.1. Что такое справочник.....	21
2.2. Простой справочник.....	23
2.3. Справочник с табличной частью.....	25
2.4. Иерархический справочник.....	27
2.5. Справочник с предопределенными элементами.....	29
2.6. Предопределенные элементы.....	30
2.7. Основная конфигурация и конфигурация базы данных.....	31
2.8. Палитра свойств.....	33
2.9. Контрольные вопросы.....	35
3-й день. Документы.....	37
3.1. Что такое документ.....	37
3.2. Документ Приходная накладная.....	39
3.3. Автоматический пересчет суммы в строках документа.....	42
3.4. Обработчик события.....	43
3.5. Одна процедура для обработки нескольких события.....	45
3.6. Документ Оказание услуги.....	47
3.7. Анализ кода с помощью синтакс-помощника.....	50
3.8. Анализ кода с помощью отладчика.....	54
Прием.....	56
4-й день. Регистры накопления.....	57
4.1. Зачем нужен регистр накопления.....	57
4.2. Что такое регистр накопления.....	58
4.3. Регистр накопления (Accumulation Register).....	60
Структура.....	60
Связь с регистратором.....	62
Уникальность записей.....	63
Регистры остатков и регистры оборотов.....	63
Агрегаты.....	64
Форма списка и форма набора записей.....	64
Функциональные возможности регистра накопления.....	65
4.4. Создание регистра накопления.....	65
4.5. Создание движений документа.....	65
4.6. Команда перехода к движениям в форме документа.....	67
4.7. Создание движений документа Оказание услуги.....	67
5-й день. Простой отчет.....	69
5.1. Что такое отчет.....	69

5.2. Создание отчета.....	69
6-й день. Макеты.....	72
6.1. Что такое макет.....	72
6.2. Создание макета документа.....	72
6.3. Редактирование макета документа.....	73
6.4. Редактирование формы.....	76
7-й день. Периодические регистры сведений.....	77
7.1. Регистр сведений (Information Register)	77
Структура.....	77
Периодичность.....	78
Подчинение регистратору.....	79
Уникальность записей.....	80
Формы.....	80
Форма списка.....	81
Форма записи.....	81
Функциональные возможности регистра сведений.....	81
7.2. Создание периодического регистра сведений.....	82
7.3. Автоматическая подстановка цены в документе.....	83
7.4. Автоматическое заполнение цены в документе ОказаниеУслуги.....	84
8-й день. Перечисления.....	86
8.1. Добавление перечисления.....	86
8.2. Изменение процедуры проведения документа.....	86
9-й день. Проведение документа по нескольким регистрам.....	89
9.1. Зачем нужно проведение документа по нескольким регистрам?.....	89
9.2. Добавление регистра накопления.....	89
9.3. Изменение процедуры проведения документа.....	90
9.4. Изменение процедуры проведения документа.....	92
10. Обратные регистры накопления.....	96
10.1 Зачем нужно создавать еще один регистр.....	96
10.2. Что такое обратный регистр накопления.....	96
10.3. Создание обратного регистра накопления.....	98
11-й день. Отчеты.....	102
11.1. Способы доступа к данным.....	102
12-й день. Бухгалтерский учет.....	125
Приложение 1. Архитектура "1С:Предприятия" как продукт инженерной мысли.....	134
Литература.....	179